A STUDY ON THE APPLICABILITY OF SYMBOLIC COMPUTATION IN STABILISING CONTROL DESIGN FOR SWITCHED SYSTEMS

AT-TASNEEM BINTI MOHD AMIN

THESIS SUBMITTED IN PARTION FULFILMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE

FACULTY OF ENGINEERING AND BUILT ENVIRONMENT
UNIVERSITI KEBANGSAAN MALAYSIA
BANGI

2012

KAJIAN TERHADAP APLIKASI KOMPUTERAN SIMBOLIK DALAM REKA BENTUK KAWALAN KESTABILAN UNTUK SISTEM PERTUKARAN

AT-TASNEEM BINTI MOHD AMIN

TESIS YANG DIKEMUKAKAN UNTUK MEMENUHI SEBAHAGIAN DARIPADA SYARAT MEMPEROLEH IJAZAH SARJANA SAINS

FAKULTI KEJURUTERAAN DAN ALAM BINA
UNIVERSITI KEBANGSAAN MALAYSIA
BANGI

2012

DECLARATION

I hereby declare that the work in this thesis is my own except for quotations and summaries which have been duly acknowledged.

4 October 2012 AT-TASNEEM BINTI MOHD AMIN

P50212

ABBREVIATIONS

A	state matrix
$\tilde{A}$	state matrix in Brunovsky form
B	state matrix
$\tilde{b}$	state matrix in Brunovsky form
C	output matrix
K	controller gain
k	controller gain in Brunovsky form
M	Lyapunov matrix using Method *M* (LMI Solver)
$M_{feasible}$	Lyapunov matrix using Method *M* (Yalmip)
M_0	Lyapunov matrix using Method *H* (cvx, Yalmip, LMI Solver, Maple)
n	switching sequence
$\mathbb{N}$	natural number
P	symmetric positive definite matrix
$\mathbb{R}$	Euclidean Space
t	time
$u(t)$	input or control vector
W	Lyapunov matrix using Method *M* (theory)
x	state vector
$\dot{x}(t)$	derivative of the state vector with respect to time
x_e	equilibrium point
$y(t)$	output vector
$\mathbb{Z}$	integer number
$\sigma(t)$	input sequence at time t

ABSTRACT

This thesis examines the problem of designing controllers for switched systems that assures stability of the overall system. The switched systems here refer to systems whose dynamic behaviour changes from time to time. Stability concepts for continuous time and discrete event systems cannot be used to assure stability of switched systems as mode switching sequence and dwell time influences the stability of the overall system. One method of ensuring stability of switched systems is by proving the existence of a Common Lyapunov Function for the system. However, finding a Common Lyapunov Function is not trivial. Most methods that have been introduced to solve this problem involve the formulation of the system dynamic model and constraints into a Linear Matrix Inequality (LMI) structure. Then, computational methods are used to solve the LMI problem. Two problems arise from using LMIs to find solutions. Firstly, available LMI solvers use numerical computation which raises the possibility of rounding off errors. Secondly, the computational burden would be quite heavy, especially if the switched system comprises of a large number of subsystems or the subsystems are of a high order. The Haris-Rogers method is an alternative approach that has been previously developed for designing controllers based on the existence of a Common Lyapunov Function. In this approach, the problem is reduced to solving two sets of Linear Inequalities (LI), hence reducing the computational burden, as compared to methods that use LMIs. To overcome rounding off errors, symbolic computation methods should be used. However, this would require more computational power compared to numerical computation methods. In this study, a Switched System Control Design Toolbox employing symbolic computation based on the Haris-Rogers solution method was developed using the Maple software. A switched system with four second order subsystems was used as a test case and the toolbox successfully found a Common Lyapunov Function and subsequently designed the controller. For comparison, an LMI based method was tested with the same switched system using Maple and also three numerical LMI solvers, namely cvx, LMI Solver and Yalmip. Only Yalmip successfully generated a correct solution, while LMI Solver generated an incorrect solution since the controller obtained was clearly unstable. Maple and cvx failed to generate any controller. The Haris-Rogers method was also tested for the same switched system using cvx, LMI Solver and Yalmip, and all three produced correct results. All computational work and testing were performed on a system running Intel Core i5 2.67 GHz, 64-bit operating system with 2 GB RAM.

ABSTRAK

Tesis ini membicarakan masalah mereka bentuk sistem kawalan untuk sistem pertukaran yang dapat menjamin kestabilan keseluruhan sistem tersebut. Sistem pertukaran yang dimaksudkan di sini merujuk kepada sistem yang perilaku dinamiknya bertukar-tukar dari masa ke semasa. Konsep kestabilan bagi sistem masa berterusan dan sistem peristiwa logik tidak dapat digunapakai untuk menjamin kestabilan sistem pertukaran. Ini kerana jujukan pertukaran mod dan masa lengah di antara pertukaran memberi kesan kepada kestabilan keseluruhan sistem. Satu cara untuk memastikan kestabilan sistem pertukaran adalah dengan membuktikan kewujudan Fungsi Lyapunov Sepunya untuk sistem tersebut. Namun pencarian Fungsi Lyapunov Sepunya adalah satu masalah yang tidak mudah diselesaikan. Kebanyakan kaedah yang telah diperkenalkan untuk menyelesaikan masalah ini melibatkan perumusan model dinamik sistem pertukaran tersebut serta kekangan-kekangannya ke dalam bentuk Ketaksamaan Matriks Lelurus (LMI). Kemudian, kaedah komputeran digunakan untuk menyelesaikan masalah LMI ini. Penyelesaian menggunakan LMI menimbulkan dua masalah. Pertama, penyelesai-penyelesai LMI yang sedia ada menggunakan komputeran berangka, yang memungkinkan berlakunya ralat penggenapan. Kedua, pembebanan yang dikenakan ke atas komputer agak tinggi, terutamanya jika sistem pertukaran itu terdiri daripada banyak subsistem atau subsistem-subsistemnya berdarjah tinggi. Kaedah Haris-Rogers adalah kaedah alternatif yang telah dibangunkan sebelum ini untuk mereka bentuk sistem kawalan berasas kepada kewujudan Fungsi Lyapunov Sepunya. Dalam kaedah ini, permasalahan dikurangkan kepada penyelesaian dua set Ketaksamaan Linear (LI), dan dengan itu pembebanan komputeran adalah kurang berbanding kaedah-kaedah yang menggunakan penyelesaian LMI. Untuk mengelakkan berlaku ralat penggenapan, kaedah komputeran simbolik seharusnya digunakan. Walau bagaimanapun, ini akan menggunakan kuasa komputeran yang lebih tinggi berbanding dengan kaedah komputeran berangka. Dalam kajian ini, satu Kotak Kekunci Reka Bentuk Pengawal Sistem Pertukaran berasaskan komputeran simbolik yang melaksanakan kaedah penyelesaian Haris-Rogers telah dibangunkan menggunakan perisian Maple. Satu sistem pertukaran dengan empat subsistem tertib kedua telah diguna sebagai kes ujian dan kotak kekunci ini berjaya mendapatkan Fungsi Lyapunov Sepunya dan seterusnya mereka bentuk sistem kawalannya. Sebagai perbandingan satu kaedah berasaskan LMI telah diuji untuk sistem pertukaran yang sama menggunakan perisian Maple dan tiga penyelesai LMI secara berangka iaitu cvx, LMI Solver dan Yalmip. Hanya Yalmip berjaya menjana keputusan yang betul. Kaedah Haris-Rogers juga diuji untuk sistem pertukaran yang sama menggunakan cvx, LMI Solver dan Yalmip dan kesemuanya menghasilkan keputusan yang betul. Kesemua kerja komputeran dan pengujianan telah dijalankan menggunakan sistem Intel Core i5 2.67 GHz dengan sistem operasi 64-bit dan RAM 2 GB.

TABLE OF CONTENT

APPENDICES

LIST OF FIGURE

LIST OF TABLE

CHAPTER I

INTRODUCTION

1.1 INTRODUCTION

Hybrid systems are dynamic systems that arise out of the interaction of continuous state dynamics and discrete state dynamics. The state variable is called discrete if it takes on a countable or finite number of values $\{q_1, q_2, \dots\}$. The discrete-time state equations are

$$x(n+1) = Ax(n) + Bu(n)$$
$$y(n) = Cx(n) + Du(n)$$

where $n \in \mathbb{N}$ or $\mathbb{Z}$. On the other hand, the state variable is called continuous if it takes values in n- dimensional Euclidean space $\mathbb{R}^n$ for some $n \geq 1$. The continuous time state equations are

$$\frac{dx(t)}{dx} = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) + Du(t)$$

which represent the next state of the system with respect to current state $x(t)$ and input $u(t)$ of the system. Hybrid systems involve both these types of dynamics (Lygeros, 2004). Switched system, which is a type of hybrid system, is one which is being given much attention to by the control systems research in this decade. The switched system is able to function by having a system that is represented by a few subsystems that are connected by a command and a certain set of conditions.

Problems of controllability, observability, converseability and stabilizability of the switched system have always been discussed. To explain the problems that occur, referring to Figure 1.1, one hybrid system is represented by four linear subsystems;

namely subsystem 1, subsystem 2 to subsystem 4. The dynamic of the subsystems can be explained by the continuous time dynamic equation (CT) while the concept of dynamic in connecting subsystem 1 to subsystem 2 can be represented by a discrete event system (DES). However, the concept of stability in CT or DES cannot be used to determine the stability of the switched system for this hybrid system. This is due to the fact that the mode-switching sequence and the dwell time between switching gives an impact to the stability of the whole system. Therefore, a new method is needed to design a control for the hybrid system as to guarantee the stability within the switched system.

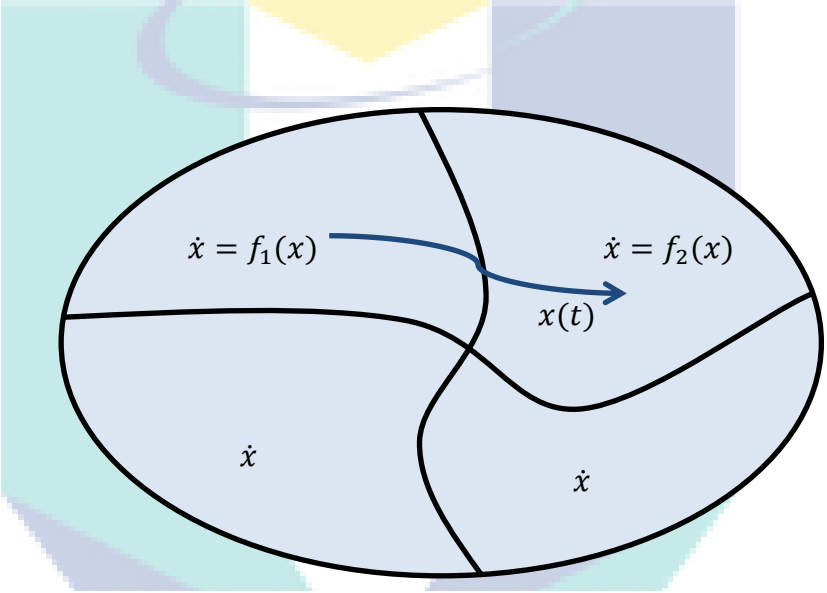


Figure 1.1 The pathway of a state variable in a hybrid system

Many methods have been suggested and discussed for this purpose. One way of ensuring the stability of the switched system is by proving the existence of a Common Lyapunov Function for the system. However, the search for a Common Lyapunov Function is a problem which is not easy to be solved. Most methods introduced to solve this problem involve the formulation of the dynamic model of the switched system along with its constraints in to the form of Linear Matrix Inequalities (LMI). Then computational programs are utilized to solve the LMI. To solve a LMI, numerical computational-based programs like optimization programs or LMI solvers may be used. For example, the LMI Solver, SeDuMi and CVX can be used to obtain the objective x in the constraints LMI $Ax \leq b$ where A is a matrix in which x and b are constants. Finding solutions using LMIs is hindered by two problems. Firstly,

existing LMI solvers utilizes numerical computing which leads to the possibility of rounding-off errors. Secondly, the computational demands are very high, especially if the switched system consists of many subsystems or if the subsystems are of a high order. The tax on computerized systems for each program may possibly create a difference in the solutions for the inequality that is sought. This error can create a doubt towards the algorithm and the inequality proposed in terms of its validity.

However, throughout this research, the solutions of the LMI that are sought have not yet been discovered by the use of symbolic computational-based programs. The use of a processor with a floating point controller may possibly make it less computationally taxing and also reduce the errors in generating solutions. Can the use of symbolic computational-based programs take place of numerical computational-based programs which all this while have been the basic program in solving the problems of stability in switched systems?

1.2 PURPOSE OF STUDY

The objective of this study was to construct a hybrid system controller design tool for switched systems based on symbolic computations. In achieving this objective, we also sought to:

- test the computational ability of a few types of LMI solvers to solve the problems of the design of control systems for switched systems.
- compare the computational taxation in solving the problems of the design of control systems for switched systems using methods based on the LMI and the method constructed by Haris et al. (2007).

1.3 SCOPE OF STUDY

The method constructed only takes into account the switching that happens internally. This means that any external disturbances like the input of external forces and disturbances or changes of weather are not taken into account in the construction of this algorithm. The changes also refer to the system in which its switching behaviours are determined by basic behaviours like due to the state variables only. The main

focus is switched systems with a single input – single output (SISO) with a second order continuous time subsystem. The algorithm is analysed by using the symbolic computational software Maple and the comparison of its computational taxation is done by using three types of numerical computational-based programs which are based on MATLAB; which are the cvx, Yalmip and LMI Solver. The controller toolbox design only takes the issues of stability into consideration while the performance of the system is not considered in this thesis. Each computerized process is handled by an Intel Core i5 processor, with a 64-bit operational system of 2.67 GHz speed with a memory (RAM) of 2.00GB.

1.4 THE SIGNIFICANCE OF STUDY

The method introduced is a new method in solving the problems of stability in a hybrid system. This method is seen to potentially reduce computational taxations due to the usage of a first order linear inequalities (LI). It may be solved by any type of solver. The interpretation of this method by using symbolic types of computations is a unique way which has never been used in any researches of control systems prior to this. This method will take over the conventional method of using numerical computations to solve algorithms constructed. This will also open up many issues with regards to handling it. The hybrid systems controller toolbox of constructed will assist researchers and students in the study of the stability of switched system since the pole-zero tests and transient response graphs for the switched systems are inserted together with the toolbox.

CHAPTER II

STABILITY OF SWITCHED SYSTEM

2.1 INTRODUCTION

Switched systems are made up of a collection of linear subsystems with rules that govern switching between these subsystems (Sun and Ge 2005). The switching law may be either supervised or unsupervised, time-driven or event driven (Ge et al. 2001). Switched systems exist in many practical applications, for example in control systems for gear transmissions, airplanes, traffic control and also for switching powers in the electric industry (Liberzon and Morse 1999).

Much research has been done with regards to switched systems. These encompasses the concept of system controllability, observability and also stabilizability of a switched systems. Given a linear time invariant system (Gopal 2003)

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) \qquad (2.1)$$

where A, B, and C are a matrix $n \times n, n \times 1$, and $1 \times n$ respectively, $x(t)$ which $n \times 1$ is a state vector, $y(t)$ and $u(t)$ is an input and output of the system. The system is said to be controllable when the input u is able to transfer the initial state $x(0)$ to another state $x(t)$ in finite time. For a switched system, the system is said to be controllable if at the initial time and state; there exists a switching sequence that leads the state vector to the final state within a finite interval. The system is also said to be observable, given any switching sequence within a finite interval, the initial state can

be determined by only using the output vector. In other words, by utilizing the output vector of the system, it is not impossible to know the behaviour of the whole system (Sun and Ge 2005). This state can be simplified as; if state vector $x \in R^n$ is controllable/reachable/unobservable at time t_0 , the x is controllable/reachable/unobservable at any arbitrary given instant of time.

Extensive research with regards to the controllability for continuous-time and discrete-time linear switched systems have been carried out by Ge et al. (2001) and Yijing et al. (2003) respectively in its geometric characterization. Even though this definition was constructed by taking into consideration of a single input only, Guangming et al. (2002) however has proven that the controllability for a multi-input system is equivalent to that of a single-input systems. It was later proved in Guangming and Long (2002) that for switched linear systems, there exists a basic switching sequence such that the controllable state set of this basic switching sequence is equal to the controllable state set of the system.

On the contrary, a different type of research was done by Vu and Liberzon (2006) who were interested in a new problem in switched systems. They focused on the invertibility problem of a pair of subsystems for continuous-time linear switched systems. With the information of the initial state and the output state, the switching signal and input state can be recovered. By introducing the concept of singular pairs for two systems in discrete and continuous states, an algorithm was presented for finding switching signals and inputs that generate a given output in a finite interval when there is a finite number of such switching signals and inputs.

However, this thesis will focus on the stability of switched systems with subsystems which comprise of continuous-time linear systems. The discussion will also only take into account switched systems with arbitrary switching.

2.2 STABILITY OF SWITCHED SYSTEMS

Stability can be defined when all of the controllable state variables have stable dynamics or if there are non-controllable state variables, then all the state variables

always remain within the boundaries of system behaviour. In the research of analysis the stability of switched systems, the scenario is that most researchers have the tendency to make conclusions of the behaviours of the switched systems without any theoretical application in finding solutions of a hybrid system.

There are studies of the stability of switched systems had begun with interpreting algorithms in the form of difference equations for continuous-time systems and differential equations for discrete-time systems. (Brayton and Tong 1979; Brayton and Tong 1980). However, the main idea in the study of stability within hybrid systems is that when a Lyapunov function can be created from each subsystem; and by adjusting it with the switching mechanisms of the system, the stability of the switched system can be achieved. Extracting from Zhu et al. (2007), taking into account of linear switched systems

$$\dot{x} = A_{\sigma(t)}x \tag{2.2}$$

where $\sigma(t)$: $[0, +\infty) \rightarrow \Lambda$ is a continuous function with $\Lambda = \{1,2, \dots, N\}$.

To ensure the stability of the switched system (2.1) with arbitrary switching, the common quadratic Lyapunov function (CQLF) is sufficient. The common quadratic Lyapunov function, $x^T Px$, with a symmetric positive definite matrix, $P > 0$ is called the CQLF to $\{A_\lambda | \lambda \in \Lambda\}$ if

$$PA_\lambda + A_\lambda^T P < 0, \ \ \forall \lambda \in \Lambda. \tag{2.3}$$

Referring to Lyapunov and Fuller (1992), the Lyapunov stability or x_e happen when all solutions for a particular dynamic system that starts near the equivalent point x_e, it will remain in that position of close proximity. The robustness of the stability is even more increased when all the solutions start near x_e approached towards x_e or also known as asymptomatically stable.

This theorem is also supported by Martin and Dayawansa (1996) for switching within finitely linear systems with arbitrary switching, in which the asymptomatic stability for all switching path is equivalent to the common Lyapunov function for all of the subsystems for all subsystem in the family. However, researchers face difficulty in finding the upper bound of degree of the system. This is following the expression of an example that shows that the Lyapunov function fails to form a CQLF. This theorem is also used by Jianhong et al. (2008) for linear time-invariant systems by using the LMI optimization approach to find the CQLF. In addition to this discovery, King and Shorten (2004) stated that for a group of stable and finite matrices $\mathcal{A} = \{A_1, \dots, A_m\}$, CQLF will not exist if and only if not all positive semidefinite matrices $X_1, \dots, X_m$ equals to zero, in which

$$\sum_{i=1}^{m} A_i X_i + X_i {A_i}^T = 0. \tag{2.4}$$

Research by Dayawansa and Martin (1999) with linear switched systems and Mancilla-Aguilar and Garc á (2000) with nonlinear switched systems with arbitrary switching have proved that each system respectively is globally uniformly asymptotically stable and locally uniformly exponentially stable with converse Lyapunov theorem. Frequently, it is seen that the focus of most studies is given to the quadratic stability of the system compared to uniformly asymptomatically stability. Quadratic stability can be achieved when the Lyapunov function is quadratic in the state variable and is independent of time. Nonetheless, the overall stability of the system is very dependent on parameters and time. A study done by Mason et al. (2007) has found that entire criterion of the stability of a system are actually equivalent in which can be rephrased as quadratic stability needs only to be tested on quadratic polytopic Lyapunov function. They then found that this definition is not applicable for discrete time switched systems. This discovery has enforced the theorem proven by Sun (2007) for continuous-time switched systems in which the uniformly asymptomatically stability does not fulfil the quadratic Lyapunov function.

As an addition to the topic of the stability behaviours within the switched systems, besides asymptomatic stability and quadratic stability, the input-to-state stability (ISS) is also an important property, mainly for nonlinear systems. Due the

fact that the system might not be overall stable despite the stability of each subsystem, ISS becomes the preserver within the system in the efforts of achieving overall stability of the whole system (Liberzon 1999). To enforce this theory, Wenxiang et al. (2001) identified the type of mode of every subsystem before the suitable controller and duration of controller usage were identified using model method to ensure that the whole system is in the ISS. Nesic and Liberzon (2005) then demonstrated the use of ISS small-gain theorems as an power extension to be used in hybrid systems.

Further reference with regards to stability within systems with impulse effects can be found in Hespanha and Morse (2002), time-varying systems in Ezzine and Haddad (1988) and Qi et al. (2005) and time-delay in Dayawansa and Martin (1999). More information with regards to the analysis of the stabilizability within switched systems in industries such as aircraft and PI controllers of vehicles with automatic transmission can be referred to in Decarlo et al. (2000) and Brockett (1993) for manual transmission.

The next section is a survey of the methods used by researchers in ensuring the stability of switched systems. As mentioned earlier, this thesis will only focus on the switching from continuous-time linear systems with arbitrary switching. The methods used are numerous and are not dependent on the usage of the Lyapunov function alone. Section 2.2.1 focuses on the method called multicontrollers while Section 2.2.2 is concerning stability through the use of the Hurwitz matrix. State transformation will be further explained in Section 2.2.3 while the use of stabilizing switching control strategy is elaborated further in Section 2.2.4. Finally, the use of Lie algebra will be discussed in 2.2.5.

2.2.1 Multicontrollers

In this method, the main aim is to construct a multicontroller system within a hybrid system. This multicontroller will be used and will switch effectively among the controllers for each subsystem by taking into account that a single controller is incapable of stabilizing the whole system with any switching sequence. Hespanha and Morse (2002) believe that if a multicontroller is chosen properly, then for every

switching between subsystems, it will be guaranteed to be uniformly exponentially stable. The system studied is the time invariant system. Each stable controller transfer matrix for every switching between subsystem will be mentioned and be represented by equations in the form of Youla parameterization. This implies that switched systems will occur through certain parameters compared to switching through a controller transfer matrix. The formation of multicontrollers in the form of the Youla parameterization must fulfil the Lyapunov function. This method of multicontrollers is also used by Stewart and Dumont (2006) for discrete time systems.

Hespanha et al. (2007) have raised a particular concern with the use of this technique. Is this technique applicable for use in switched systems that possess an interconnection between the feedback loop and the multicontrollers? If it is applicable, then how do we obtain the initial state with the controller after it has transformed back into the feedback loop? Consequently, they introduced the norm-constraints in the optimization of the state-reset in which it will produce a transient response which also is a preserving the (input to state) stability of the system.

However, an argument raised by Blanchini et al. (2008) said that the system which uses Youla Parameter could not be arbitrary because it is already constructed. Therefore, he introduce an extended controller device to the system with the Youla Parameter, called the switching compensator (consists of observer and a (dynamic) state feedback) as in Figure 2.1 which is supported by a polyhedral Lyapunov function which is based on the separation principle to fix the problem. However, all of them cannot provide the bound for the system order and it is highly computational-demanding.

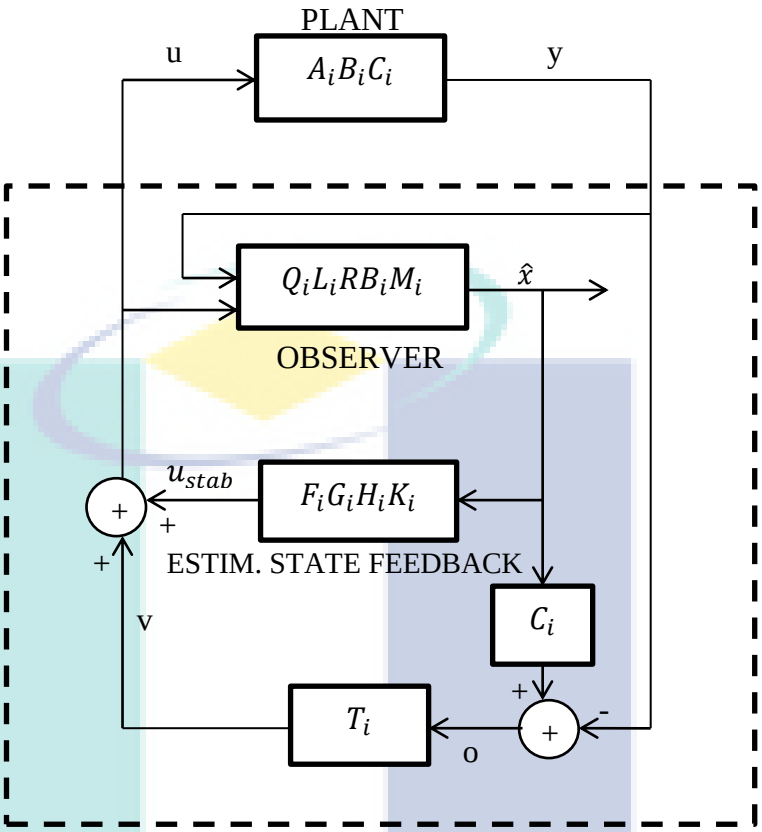


Figure 2.1 Switching Compensator

Source: Blanchini et al. 2008

2.2.2 Hurwitz Stability

Referring to equation 2.2 with regards to guaranteeing stability for switched systems within linear systems, the quest for finding the existence of a Lyapunov function is a conservative way of finding stability.

Besides this method, Mason and Shorten (2003) conjectured that asymptomatic stability can be achieved in a positive linear system with arbitrary switching by testing the Hurwitz stability of the convex hull of a Metzler matrix set. The Hurwitz matrix is a square structured matrix with $n \times n$ which is built together with a constant in a particular polynomial as follows

$$H(p)=\begin{bmatrix} a_1 & a_3 & a_5 & a_7 & \cdots & 0 \\ a_0 & a_2 & a_4 & a_6 & \cdots & 0 \\ 0 & a_1 & a_3 & a_5 & \cdots & 0 \\ 0 & a_0 & a_2 & a_4 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 \end{bmatrix}$$

A matrix $H(p)$ is called a Metzler matrix if its off diagonal elements are non-negative (Narendra and Shorten, 2010). The Hurwitz stability is confirmed when all the eigenvalues of $A + BK$ designed from $u(t) = Kx(t)$ are in the open left complex plane (Zhai et al., 2006). According to Zhang (2008), there are two criteria to test the stability of Hurwitz. One of the indirect methods is by testing the eigenvalue of the matrix including computing its Jordan canonical function, calculating the invariant factors, etc. However, it is not an easy task to complete these computations due to the computational complexity. Another method deals with the stability based on the entries of a given matrix directly.

This conjecture can be true for the system which is constructed only from a pair of second order Metzler matrices and for the system which is constructed from the arbitrary finite number of second order Metzler matrices, while the conjecture is generally false of higher order systems (Gurvits et al. 2007). This assumption is also taken up by Guisheng et al. (2006) for continuous-time systems which form an algorithm based on the Lie Algebra for which the system is exponentially stable with arbitrary switching. The Lie Algebra method will be discussed in detail in section 2.2.5.

2.2.3 State Transformation

The state variable transformation method is a way of changing the initial state variable of the system to a new state that is capable of creating a stabilizing strategy for that system before the new state is changed back to its initial state. Davrazos and Koussoulas (2002) utilized the canonical form in canonical coordinates as a medium for searching for the stabilizing strategy for switched linear systems. The problem of

stability for this system is interpreted by using a state feedback control in canonical composition. Then, a state estimator is introduced within the system to estimate the state variable to be sent to the feedback loop and to achieve the algorithm of the overall system before it is changed back to its initial state.

The main objectives are to find the suitable control input and switching laws which guarantee the system will be uniformly asymptomatically stable. Before the transformation process, it has to be made sure that the system is controllable and observable. The system is stabilizable only if the state space matrices are Hurwitz or equivalent, the unstable mode of state space matrices is controllable. Further reading with regards to state transformation can be referred to in Geng (2010).

a. Linear state transformation

Li et al. (2001) instead introduced a linear state transformation to find a stable convex combination for a class of switched systems. The linear state transformation will decompose each subsystem into stable and unstable parts; for which with each stable part, there naturally exist a Lyapunov function. Under some conditions imposed on the original switched system, the sum of these Lyapunov functions is shown to a Lyapunov function for a convex combination of the whole switched system.

b. State-switched Transformation using Differential Petri Nets

Differential Petri Nets (DPN) is a simulation software in which it is the sequel for another simulation software called “Hybrid Petri Nets” which is used for stabilize switched linear systems (Davrazos and Koussoulaz 2007). Petri Nets is a basic hybrid system controller configuration to control continuous switching transformation from initial discrete event to output discrete event form continuous input signal and output signal (Moor et al. 2006). The state transformation introduced in Petri Nets to exploits the capability and the advantages of continuous type Petri Nets models in representing continuous varying quantities in a discrete event setting by making use of simulation mechanism. The stability analysis by the novel transformation of the equation in the DPN of discrete event system compared with the state switched system in a DPN framework using switching hyperplanes was presented. Stability condition was performed and expressed in Linear Matrix Inequalities (LMI).

The analysis of stability in state-switched hybrid systems with state transition using DPN has been divided in two parts. The first part is to model the state-switched hybrid system by switching hyperplanes in the DPN framework and the second part is to transform the DPN fundamental equation into a linear switched system form.

Figure 2.2 is showing the model of the subsystem in the DPN working plane form. Transition between two different subsystems is expressed in the form of i and j as $f_{ij}^{T}x(t) = 0$ is modelled in the DPN framework. The differential place P_{DF1} and P_{DF2} are tested for the expression $ax_1 + bx_2 = 0$. Figure 2.3 instead shows the example of the DPN framework that have been formed into subsystems $i$and j. Further reading of this method can be found in Davrazos and Koussoulas (2002).

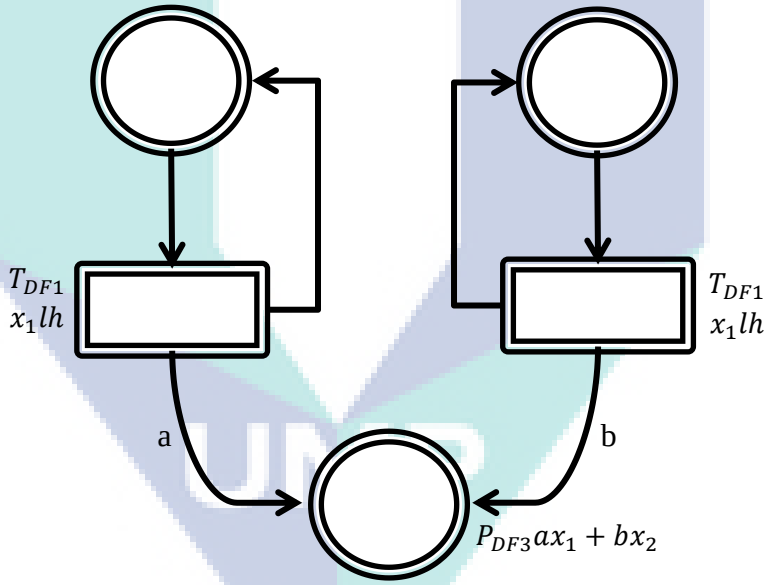


Figure 2.2 Modelling of hyperplanes in DPN framework

Source: Davrazos and Koussoulas 2007

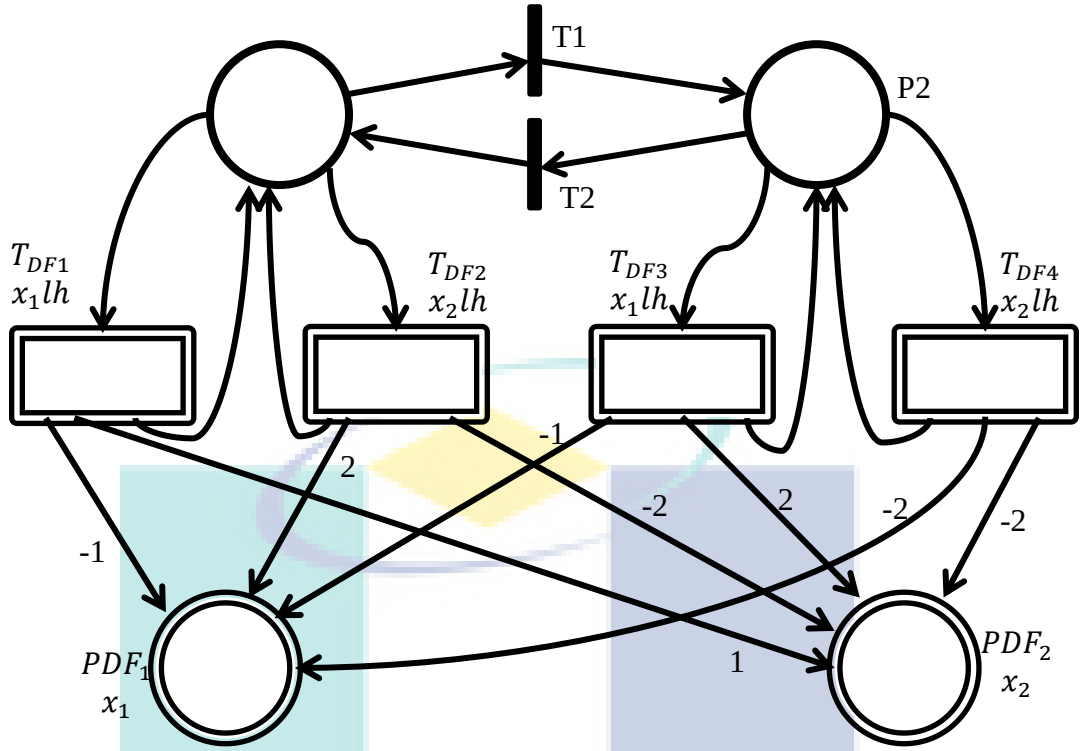


Figure 2.3 Representation of a switched system in a DPN framework

Source: Davrazos and Koussoulas 2007

2.2.4 Stabilizing switching control strategy

Montagner et al. (2006) on the other hand, provided a convex design method for switching feedback gain for switched linear systems with arbitrary switching. A quadratic Lyapunov function with a common matrix is used to derive a stabilizing switching control strategy that guarantees, for any arbitrary switching rule:

(1) The location of the poles of each linear subsystem of a continuous-time switched linear system is inside a chosen circle in the left-hand half of the complex plane.

(2) A minimum disturbance attenuation level for the closed-loop switched system.

The features mentioned above are said to be important because for the first one, it will improve the dynamic response assigning bounds for overshoot, settling time and frequency of oscillation. It also will be able to ensure the robustness of the switched system facing disturbance that are energy signals. The sufficient LMI design

conditions based on quadratic Lyapunov functions with a common matrix with very low numerical complexity is presented. This LMI condition is allowed to determine switched state feedback gains that stabilize the closed loop system including pole location and robustness of the system.

The stability of the closed-loop switched system with γ disturbance attenuation level and the pole location of each linear subsystem inside the circle $\mathcal{C}(d_j, r_j)$ is visualized in Figure 2.4 as shown:

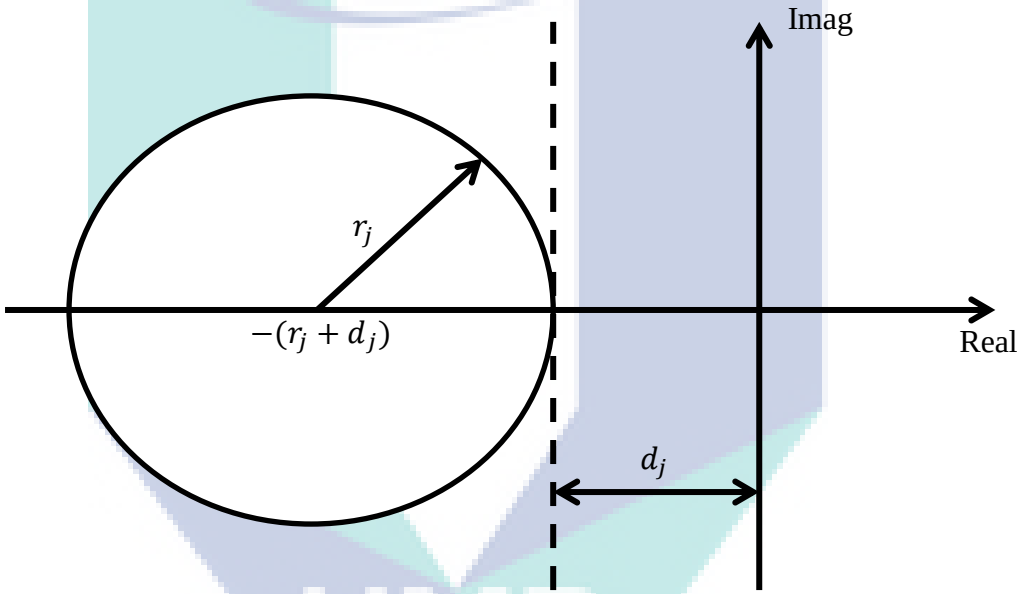


Figure 2.4 Circular region $\mathcal{C}(d_j, r_j)$ for pole location

Source: Montagner et al. 2006

2.2.5 Lie Algebra

Another popular method in the study of the stability of switched systems is by using the Lie Algebra that is derived from a stable linear system. Lie Algebra is an algebraic structure that usually used when learning of geometrical objects. Liberzon et al. (1999) conjectured that if the Lie Algebra which is generated from matrices of stable linear time invariant systems is nilpotent (which means that the Lie brackets of sufficiently high order equal zero), then the system is asymptotically stable for any

switching signal. They have proved their conjecture to be true for two subsystems for the third-order Lie brackets. The Lie bracket meant here is the vector space in a particular graphic space in a binary operation.

Further research has been done by Agrachev and Liberzon (1999) discussed the subject by showing that the arbitrary switching system will be exponentially stable if the Lie algebra is solvable. Zhu et al. (2007) however claims that most methods for achieving stability by using the Lie Algebra is not solvable. Hence, they proved that by combining the Lie Algebra with the N-B structure and also the CQLF; a new mathematical structure can be formed to offer the solution for the stability problem in which was unachievable through previous studies.

2.3 HARIS' METHOD

Haris et al. (2007) was suggested to guarantee the quadratic stability of a switched system with consisted of second order subsystems. The aim is to find the existence of a set of feedback control law $u_i = K_i x$ that share CQLF, $x^T M x$ in the subsystems. The subsystem is transformed to Brunovsky controllable canonical form to find the existence of CQLF. If exist, it guarantees the stability of the switched system. The feedback controller, K_i from a feedback control law can be found by solving two set of linear inequalities (LI) in the original plane. LI is the $x \leq b$ where x and b is a variable. The stability of a switched system is proven when the Lyapunov function is negative definite which defining M as common Lyapunov function. The Lyapunov function is said to be negative definite if all its eigenvalues have negative solution.

2.4 DISCUSSION

The stabilizability of switched systems can be achieved by various methods. Among the methods that can be used are mathematical and geometrical methods, the use of simulation softwares or the implementation of strategies over the system. Even though the method shown is focused on continuous linear systems with arbitrary switching, this method can be further explored in much detail according to the types of systems studied.

Each system studied has been interpreted into mathematical equations or inequality forms. The mathematical equations are then used to prove the algorithms created. A few questions do arise from this discussion. Will the equations and inequalities be solvable if the algorithms were to be applied for use in industries and the education field later on? Will computerized calculations be able to solve the problems faced? And what is the most suitable software to actually solve this problem?

The method using Youla parameterization is highly computation demanding. Determination of Hurwitz stability by solving the eigenvalues in Jordan canonical function is complicated to be performed by computerized system. Hence, it has been proved by Zhang (2008) that it failed to be used for higher order system. Furthermore, finding stability using state transformation method by Davrazos and Koussoulas (2002) can only be solved if the state matrix is Harwitz. Most Lie Algebra cannot be solve unless combined it with N-B structure and CQLF. On the other side, Montagner (2006) used a simple mathematic to find CQLF which guarantee the stability of a switched system. In short, most suggested method are actually very computational demanding in solving the problems.

As a solution to this problem, a research will be tested on its computational demanding to find the feedback controller that guarantees the stability of a hybrid system. A new algorithm will then be introduced in this thesis in which it is guaranteed less computational demanding. This algorithm is described using LI and optimization using a software based on symbolic computational-based software. The algorithm using symbolic computational-based will be described in detail in Chapter IV. A hybrid system controller design toolbox is created and constructed to realize and apply this algorithm to obtain a controller for each subsystem to ensure the stability of the whole system. This toolbox will also demonstrate the dynamic response for a hybrid system.

CHAPTER III

COMPUTING CAPABILITY TEST FOR METHOD *M*

3.1 INTRODUCTION

In the previous chapter, various methods have been discussed and presented to find the stabilising controller for various types of switched systems. For all problems posed, the solution obtained were based on finding the existence of a common Lyapunov function (CLF) or a multiple Lyapunov function (MLF), besides a few other solutions that have been suggested. For instance, Montagner et al. (2006) introduced a convex design method to find the switching feedback controller using the CLF while Blanchini and Savorgnan (2006) instead insist that the stability of a switched system cannot be proven with the existence of the converse Lyapunov theorem. Jin and Brown (2010) reviewed the use of the MLF in the search of switching signals for an unstable switched system with arbitrary switching sequences.

Normally, each CLF and MLF problem explained will be included in a set of LMI. An example of an LMI suggested for ensuring the stability of a switched system using various methods can be referred to in the study done by Sun and Ge (2005). However, to achieve the solution for the LMI and furthermore ensure the stability of the hybrid system, very high computing powers are needed. For this matter, in this chapter, a study from Montagner et al. (2006) is used to test the computing capability that is needed to solve the suggested LMI. A switched system with four subsystems will be used as a case study. Numerical computational-based programs such as cvx, Yalmip and the LMI Solver from Matlab and symbolic computational-based programs like Maple will be used. If there is a solution, the feedback controller obtained will be used to prove the existence of a negative definite CLF within that switched system.

3.2 PROBLEM STATEMENT

Consider a continuous-time switched system in which the subsystem N is written as

$$\dot{x} = A_i x + B_i u \tag{3.1}$$

in which $x \in R^2$, $u \in R$, $A \in R^{2\times 2}$, $B \in R^{2\times 1}$ and $i = 1, \dots, N$.

The problem is to ensure the validity of the switch control conditions

$$u(t) = K_i x \tag{3.2}$$

in which a switched system with a closed loop $\dot{x} = (A_i + B_i K_j)x(t)$ is asymptotically stable.

Definition 1 (Montagner et al. 2006)
If the symmetric positive definite matrix $W \in R^{n\times n}$ and the matrix $Z_i \in R^{m\times n}$, $i = 1, \dots, N$ exist, in which

$$A_i W + W A_i^T + B_i Z_i + Z_i^T B_i^T < 0, \qquad i = 1, \dots, N \tag{3.3}$$

hence the switching control condition (3.2) with

$$K_i = Z_i W^{-1}, i = 1, \dots, N \tag{3.4}$$

guarantees the stability of the closed loop within the switched system. Therefore the focus of the problem is to achieve the solutions for the matrices W, Z_i and K_i that fulfil the mentioned criteria.

Lyapunov matrix, W is defined as the CQLF, $v(x) = x^T P x$, $P = W^{-1}$ for the switched system mentioned in (3.1) by affirmation of the following Lyapunov function

$$(A_i + B_i K_i)W + W(A_i + B_i K_i)^T < 0, \;\; j = 1, \dots, N \tag{3.5}$$

with K_i given by (3.4).

For the purpose of ease for writing and explanation, this method suggested by Montagner et al. (2006) will be called the Method M in the following explanations.

Refered to Boyd (2009), the LMI (3.3) is equivalent to the LMI

$$diag(A_1W + WA_1^T + B_1Z_1 + Z_1^T B_1^T, A_2W + WA_2^T + B_2Z_2 + Z_2^T B_2^T, \dots, A_iW + WA_i^T + B_iZ_i + Z_i^T B_i^T) < 0$$

or

$$\begin{bmatrix} A_1W + WA_1^T + B_1Z_1 + Z_1^T B_1^T & 0 & \dots & 0 \\ 0 & A_2W + WA_2^T + B_2Z_2 + Z_2^T B_2^T & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & A_iW + WA_i^T + B_iZ_i + Z_i^T B_i^T \end{bmatrix} < 0.$$

To solve this LMI, schur complement method has to be used.

3.3 TESTING THE COMPUTERIZATION CAPABILITY

Consider a continuous-time switched system as in the equation (3.1) called the switch system X with four subsystems.

$$\begin{aligned} A_1 &= \begin{bmatrix} -4 & -2 \\ 9 & 5 \end{bmatrix} \quad B_1 = \begin{bmatrix} -1 \\ 2 \end{bmatrix} \\ A_2 &= \begin{bmatrix} 2 & -\frac{1}{3} \\ -3 & 0 \end{bmatrix} \quad B_2 = \begin{bmatrix} 5 \\ -6 \end{bmatrix} \\ A_3 &= \begin{bmatrix} -7 & -6 \\ \frac{19}{2} & 8 \end{bmatrix} \quad B_3 = \begin{bmatrix} 6 \\ -7 \end{bmatrix} \\ A_4 &= \begin{bmatrix} -12 & -11 \\ 13 & 12 \end{bmatrix} \quad B_4 = \begin{bmatrix} -7 \\ 8 \end{bmatrix} \end{aligned} \tag{3.6}$$

This system is used as a benchmark case to compare the computing capability for different solvers.

LMI (3.3) is constructed by guaranteed the stability for arbitrary switching sequences with linear modes $i = 1, \dots, N$, hence the simulation to obtain the transient response has been set so that the switching is in the form of an onward sequence; in which switching occurs from subsystem one to subsystem two to subsystem three and so on without violating the originality conditions formed.

3.3.1 cvx

cvx is an extra programming system that is specially designed to model disciplined convex programming (DCP). The DCP meant here is that, for every command that is run, it must fulfil the conditions that have been set called the "DCP rule set"; in which it is created based on the basic principles of convexity analysis. cvx needs to be supported by a Matlab algebraic computing system of the version 6.5.2 or above. Computing systems from Matlab commands can easily be changed into DCP commands due to the fact that the specifications used by DCP are from the basic operations of Matlab. It can solve basic optimizing problems such as linear programming (LP), nonlinear programming (NLP) and quadratic programming (QP) by using the simplest method. If the optimization succeeds, the cvx specification will be changed into Matlab commands and it can be used for the subsequent Matlab commands as normally (Grant and Boyd 2011).

To solve this Method *M*, five input variables are introduced followed by the LMI for the four subsystems that are to be solved; and this is as follows:

```
cvx_begin
variable W(2,2)
variable Z1(1,2)
variable Z2(1,2)
variable Z3(1,2)
variable Z4(1,2)
```

```
subject to
A1'*W+W*A1+B1*Z1+Z1'*B1'<=0;
A2'*W+W*A2+B2*Z2+Z2'*B2'<=0;
A3'*W+W*A3+B3*Z3+Z3'*B3'<=0;
A4'*W+W*A4+B4*Z4+Z4'*B4'<=0;
-W<=0;
cvx_end
```

Referring to the DCP that has been set, for every convex equation, each of the convexity scalar must start from the power $p \geq 1$ and $\neq 3, 5, 7, 9, \ldots$. Hence, based on the DCP, the LMI above can be solved. The output of the test run is as follows:

```
Homogeneous problem detected; solution determined
analytically.
Status: Solved
Optimal value (cvx_optval): +0

W =
   All zero sparse: 2-by-2
Z1 =
     0     0
Z2 =
     0     0


Z3 =
     0     0
Z4 =
     0     0
```

The output shows that the solution can be determined by analysis. This is proven by the output Optimal value (cvx_optval): +0. If the opposite happens, the output Optimal value (cvx_optval): +Inf will be shown. However, the LMI has failed to be solved due to the fact that the cvx unfortunately has been designed to solve only

minimum and maximum optimisation problems only. If the objective function has not been pre-set in the cvx commands, hence the commands are to determine the existence of feasible solution for that particular LMI only. This constraint leads to the failure to achieve the feedback controller for the switched system *X*. Refer to Appendix A.1 for full solution using cvx.

3.3.2 LMI Solver

The LMI Solver is one of the packages that are provided by Matlab through the LMI Lab package in the Robust Control Toolbox for solving three types of basic LMI problems which are solving LMIs, solving the minimum value for an LMI constraint and announcing the minimum Eigenvalue from an LMI constraint. All variables and inequality equations will be introduced in a structure form by using the graphical user interface (GUI) called the "LMI Editor" as in Figure 3.1 before it is solved in the Matlab command space. The use of the C-MEX processor has been especially prepared to manage the problems from the "Robust Control Toolbox". The C-MEX which contains MEX-files will connect the subroutine functions C, C++ and Fortran with Matlab if those subroutines are used in Matlab. The use and the solving of the "`for-loop`" function for the LMI form algorithm can be executed in a more structured manner and more quickly with the use of the MEX-files.

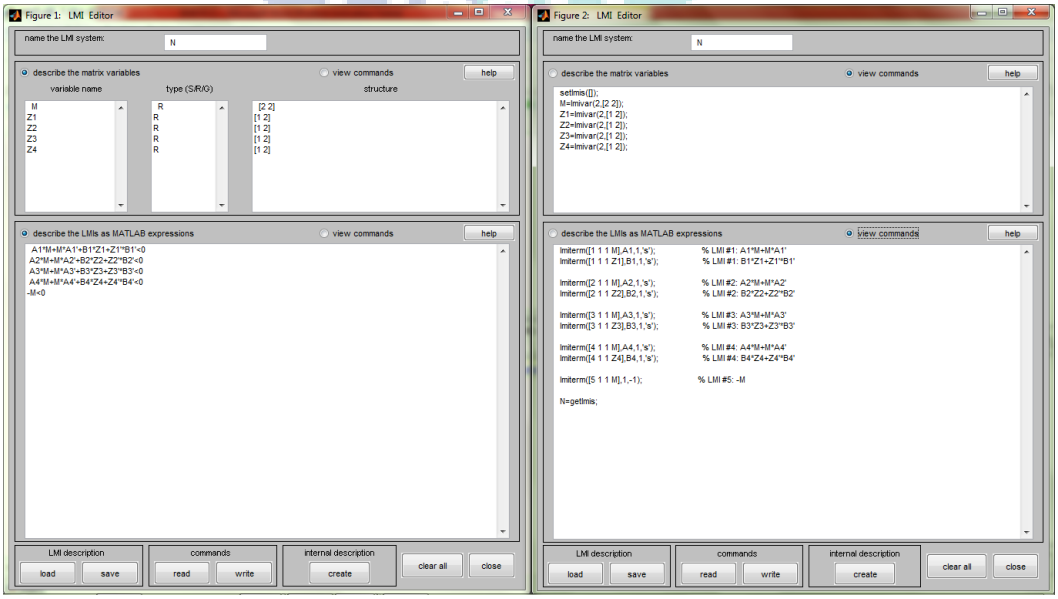


Figure 3.1 The LMI Editor

The LMI which is structured using the LMI Editor is moved to the LMI Solver to be solved as follows;

```
setlmis([]);
 M=lmivar(2,[2 2]);
 Z1=lmivar(2,[1 2]);
 Z2=lmivar(2,[1 2]);
 Z3=lmivar(2,[1 2]);
 Z4=lmivar(2,[1 2]);

 lmiterm([1 1 1 M],A1,1,'s');
 lmiterm([1 1 1 Z1],B1,1,'s');
 lmiterm([2 1 1 M],A2,1,'s');
 lmiterm([2 1 1 Z2],B2,1,'s');
 lmiterm([3 1 1 M],A3,1,'s');
 lmiterm([3 1 1 Z3],B3,1,'s');
 lmiterm([4 1 1 M],A4,1,'s');
 lmiterm([4 1 1 Z4],B4,1,'s');
 lmiterm([5 1 1 M],1,-1);

 [tmin,xfeas]=feasp(N);
M=dec2mat(N,xfeas,M);
Z1=dec2mat(N,xfeas,Z1);
Z2=dec2mat(N,xfeas,Z2);
Z3=dec2mat(N,xfeas,Z3);
Z4=dec2mat(N,xfeas,Z4);
```

For the switched system X, the LMI Solver is able to find the solution for the LMI. The five variables obtained can be used to acquire the feedback controller as mentioned in the conditions for switching control (3.4). The feedback controller for the linear system (3.1) is as follows;

Table 3.1 Feedback controller using the LMI Solver

Subsystem	**Feedback Controller, K_i**
1	[-24.859349377217598 -19.815162604854265]
2	[4.049560670684983 3.870243552564773]
3	[-3.610516536632684 -3.054792134849636]
4	[-2.593123038571612 -2.330996220943554]

Lyapunov matrix is found using LMI Solver as

$$M = 10^3 \begin{bmatrix} 6.474263251043679 & -7.858962346491398 \\ -7.603631937065424 & 9.273498841521807 \end{bmatrix}.$$

To confirm the validity of the existence of the CLF which stabilizes the hybrid system X using Method M, Lyapunov function (3.5) is utilised to obtain the eigenvalue. The stability is guaranteed if the eigenvalues of Lyapunov function have negative solution.

Subsystem 1

```
eig(M*(A1+(B1*K1))'+(A1+(B1*K1))*M)
ans =
     -0.035624850184896e+004
     -1.768638755871335e+004
```

Subsystem 2

```
eig(M*(A2+(B2*K2))'+(A2+(B2*K2))*M)
ans =
     -0.001832713694448e+004
     -1.457155689081830e+004
```

Subsystem 3

```
eig(M*(A3+(B3*K3))'+(A3+(B3*K3))*M)
ans =
      0.001175834210283e+004
      1.231520367147503e+004
```

Subsystem 4

```
eig(M*(A4+(B4*K4))'+(A4+(B4*K4))*M)
ans =
     -0.010496481177116e+003
     -6.088499374290110e+003
```

The results above clearly show that for subsystems 1, 2 and 4 the Lyapunov function, $M(A+BK)^T+(A+BK)M$ have negative eigenvalues. Hence, the stability is proven for these subsystems. However, for the subsystem 3, stability is not proven as the eigenvalues are positive. Therefore, stability is not proven for the overall system. The full program can be referred to in the Appendix A.2.

3.3.3 Yalmip

"Yet Another LMI Parser" or Yalmip is a computer command with a higher algorithm that depends on external solvers like SeDuMi, BNB and CUTSDP to solve convexity and nonconvexity optimisation problems. Yalmip functions as one of the toolboxes in Matlab. To solve a particular problem, Yalmip will suggest and choose the best solver that will fulfil the demands of the problem. However, users are allowed to choose which solver they wish to use to solve a particular problem at hand.

When using the Yalmip, an important matter that needs to be emphasized on is the introduction to each problem. The Method *M* is an LMI of a linear programming type. However, if the Yalmip used does not contain the solver for the linear programming used, the problem will then be switched to a problem from the semidefinite programming. Referring to the Yalmip program developer Löfberg (2011), there are two categories of solvers such as CSDP, DSDP, LOGDETPPA, SDPA and SeDuMi-1.3 that can be utilised for free by users and also commercialised solvers such as LMILAB, PENBMI and PENSCP. LMILAB is an internal solver commercialized by Mathworks. LMILAB is said to be less efficient to solve any basic problem but it is become very efficient if the problem is structured clearly in Yalmip structure. Therefore, Yalmip suggested LMILAB to be used to solve LMI (3.3). The LMIs defined follows;

```
M=sdpvar(2,2,'symmetric');
Z1=sdpvar(1,2);
Z2=sdpvar(1,2);
Z3=sdpvar(1,2);
Z4=sdpvar(1,2);

F=[(A1*M+M*A1'+B1*Z1+Z1'*B1'<0),(A2*M+M*A2'+B2*Z2+Z2'*B2'
<0),(A3*M+M*A3'+B3*Z3+Z3'*B3'<0),
(A4*M+M*A4'+B4*Z4+Z4'*B4'<0),(-M<0)];

solvesdp(F);
```

The five constant variables that are needed can be acquired by the following program;

```
Z1_feasible=double(Z1);
Z2_feasible=double(Z2);
Z3_feasible=double(Z3);
Z4_feasible=double(Z4);
M_feasible=double(M);
```

The feedback controllers for the four subsystems that are sought after can be acquired by the following program;

```
K1=Z1_feasible*inv(M_feasible)
K2=Z2_feasible*inv(M_feasible)
K3=Z3_feasible*inv(M_feasible)
K4=Z4_feasible*inv(M_feasible)
```

The feedback controller for the hybrid system *X* using the Method *M* using Yalmip is detailed in the Table 3.2 below:

Table 3.2 The feedback controller from the Method M using Yalmip

Subsystem	Feedback Controller
1	[-42.498387524630857 -33.971488527868246]
2	[6.131744340011608 5.496870094902704]
3	[-5.052901199286849 -4.141577120035931]
4	[-2.408226482848733 -2.158760354899222]

Lyapunov matrix is calculated as

$$M_{feasible} = 10^2 \begin{bmatrix} 3.795014078598726 & -4.578381133091899 \\ -4.578381133091899 & 5.539646919034535 \end{bmatrix}.$$

The eigenvalue obtained for every subsystem is detailed as below;

Subsystem 1

```
eig(M_feasible*(A1+(B1*K1))'+(A1+(B1*K1))*M_feasible)
     ans =
          -1.788622818453521e+002
          -0.242187329217442e+002
```

Subsystem 2

```
eig(M_feasible*(A2+(B2*K2))'+(A2+(B2*K2))*M_feasible)
     ans =
          -1.788622818516506e+002
          -0.002964983988222e+002
```

Subsystem 3

```
eig(M_feasible*(A3+(B3*K3))'+(A3+(B3*K3))*M_feasible)
     ans =
          -1.788622818384629e+002
          -0.002499399799944e+002
```

Subsystem 4

```
eig(M_feasible*(A4+(B4*K4))'+(A4+(B4*K4))*M_feasible)
      ans =
            -1.788622818594694e+002
            -0.002816654155023e+002
```

The eigenvalue for every subsystem give the negative solutions, showing that the stability is proven for the switched system. The full solution for Method *M* using Yalmip can be referred in Appendix A.3.

3.3.4 Maple

The LMI (3.3) is also tested by using a software that is based on symbolic computational-based which is the Maple. This computation manipulates the mathematical expressions to reduce errors. Therefore, it may be capable of solving the LMIs. Figure 3.2 and Appendix A.4 demonstrates the LMI that utilised Maple to find its solution. However, the LMI failed to achieve a solution after 104.78 seconds after using 1188.21 bytes of computer memory. This result clearly shows that this LMI requires much higher levels of computing powers to achieve its solution.

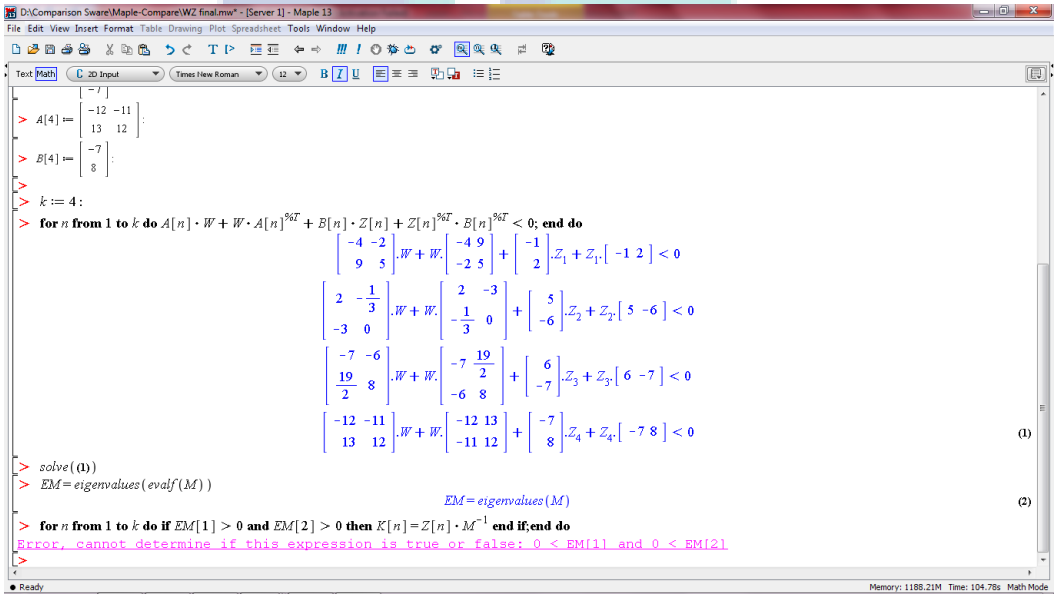


Figure 3.2 The Method *M* using Maple

3.4 DISCUSSION

The LMI (3.3) from the Method M is a sufficient condition to ensure the stability of a switched system in a closed loop. The existence of the CQLF is ensure when the eigenvalues of Lyapunov function (3.5) is negative. Three types of numerical computational-based programs and one type of symbolical computational-based program are chosen and are used to test their computing powers to solve the LMI from the Method M. The solution shows that the LMI has failed to be solved using the cvx due the limited cvx regulations. The mixed positive and negative solutions of eigenvalue of subsystems have led to the instability of the overall hybrid system X when the LMI Solver is used. The use of Maple has clearly shown that the LMI requires very highly computational demanding.

The negative definites are obtained when Lyapunov function is solved using Yalmip. Lyapunov matrix is defined CQLF by using Yalmip. The transient response using feedback controller as in Table 3.2 is plotted. The LMI is formed for switched systems with arbitrary switching. To analyse the changes in the hybrid system X without feedback controllers and with the feedback controllers from the Method M, qualitative techniques can be used; in this case the identification of the pole-zero location. The pole from the transfer function for a system is a "Laplace transform" constant variable or s that causes the transfer function to approach infinity. Figure 3.3 illustrates the pole-zero location for the hybrid system X without and with feedback controllers from the Table 3.2. This is followed by Figure 3.4 which demonstrates the transient response for the switched system X with the subsystems switching; occurring at time intervals of 20 seconds, 40 seconds, 60 seconds and 80 seconds.

In Figure 3.3, the location of the pole has changed from the right side to the left side of the real plane σ which shows that the switched system has changed from being unstable to being stable. However, the pole-zero location in every subsystem is not guaranteed the stability of the switched system X. The mode switching sequence and dwell time influences the stability of the overall system. Therefore the pole locations cannot be used to defined CQLF in continuous time system X. Even the subsystems stable individually, its does not means the whole system is stable.

The rounding-off errors throughout the calculation may cause the mixed positive and negative solutions of eigenvalues in system X by using LMI Solver. The huge different of the final result between two programs proved that there are very high rounding-off errors when using numerical computational-based programs. The computation burden leads to inability to find the validity of the method.

To avoid problems of very high demands in computing power in solving of the LMIs to ensure stability of the switched system, a simpler method should be created. Any type of software based on numerical or symbolic computation can be used to solve the algorithm, and hence, determine its stability.

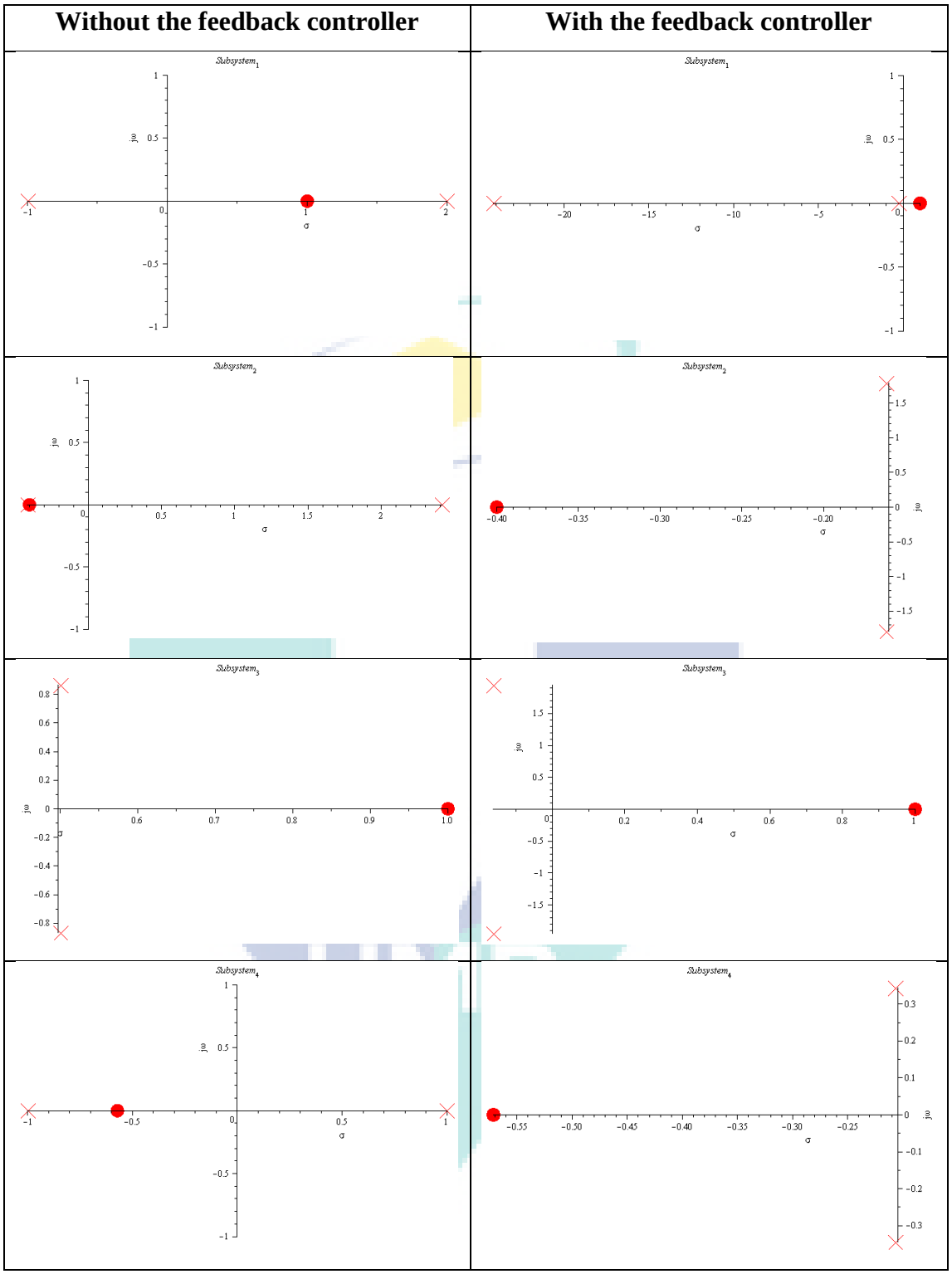


Figure 3.3 Location of the pole-zero without the feedback controller and with the feedback controller from the Method *M* using Yalmip

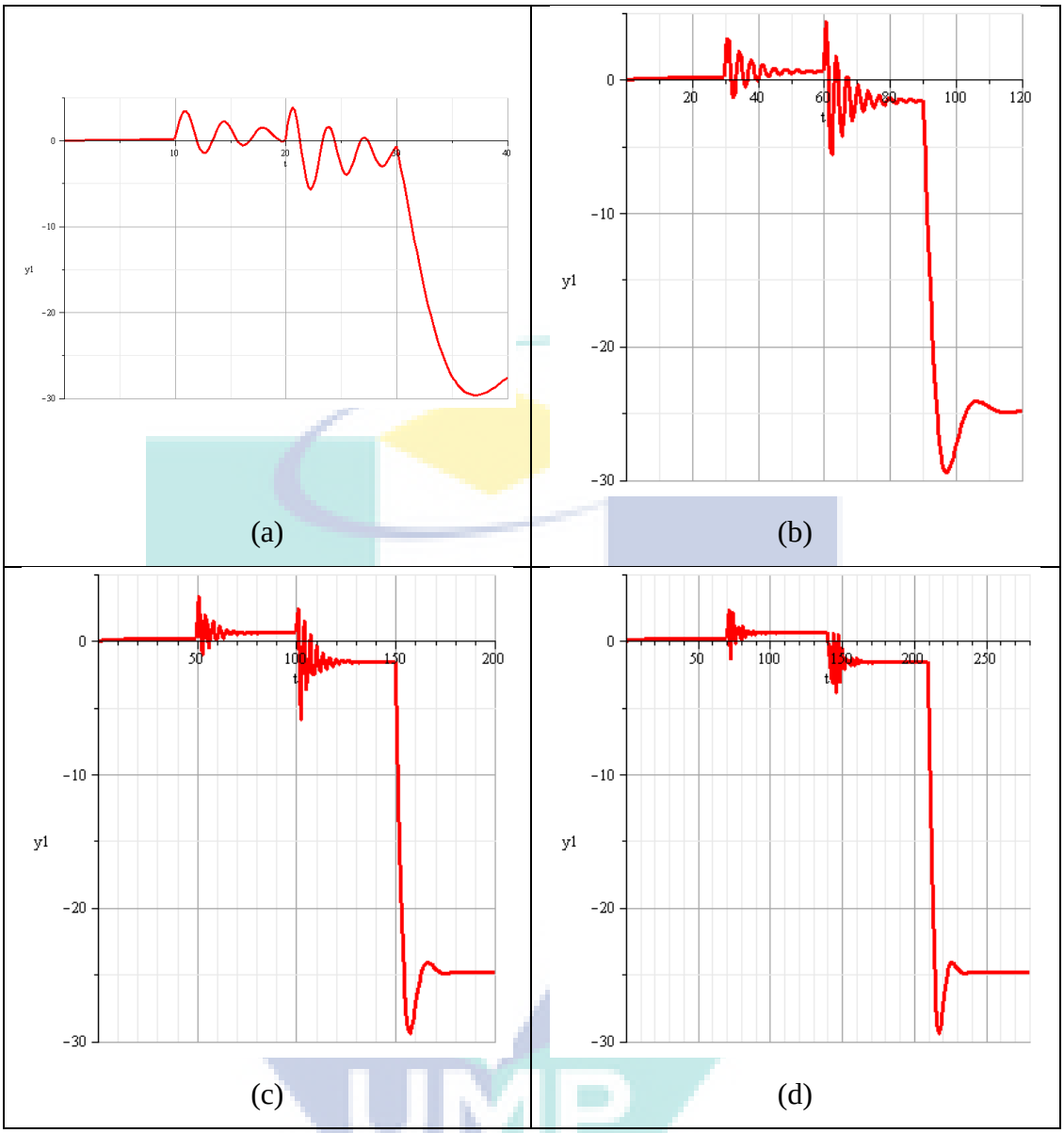


Figure 3.4 The transient response for the time intervals of exchange for every (a) 10 seconds, (b) 30 seconds, (c) 50 seconds, and (d) 70 seconds

CHAPTER IV

THE SYMBOLIC COMPUTATION BASED DESIGN TOOLBOX

4.1 INTRODUCTION

In previous studies, mostly methods that were suggested for finding the feedback controllers which guarantee stability were based on solving LMIs. In Chapter III, a previous method was presented. It is based on formulating the problem in LMI form to find the existence of the CQLF and consequently, obtain the feedback controller for the hybrid system X.

In Haris et al. (2007), a different method has been suggested to find the feedback controller that can guarantee the stability of the overall switched system. This method reduces the burden on the computation process as it seeks to solve two sets of LI as compared to solving only one set of LMI. Haris and Rogers (2008) then interpreted the algorithm into a numerical computation program called the "MATLAB Toolbox". However, the use of rounding errors in the calculations in the development of a toolbox using numerical computation programs leads to the issue or problem of inaccuracy for certain cases.

To overcome this problem, in this chapter, the algorithm will be interpreted into a symbolic computation form through a hybrid system feedback controller design with the intention to avoid the inaccuracy that occurs with the use of numerical computation. The focus is centred on the search for the existence of a common Lyapunov function (CLF) within the algorithm; in this which would guarantee the stability for switched systems with arbitrary switching. The switched system of

discussion is limited to the type of a single input single output (SISO) with a second order continuous time linear subsystem.

4.2 THE HARIS' METHOD

For the purpose of ease for writing and explanation, this method suggested by Haris et al. (2007) will be called the Method H in the following explanations. The hybrid system with N subsystems considered in Method H is as follows

$$\dot{x} = A[n]x + B[n]u \tag{4.1}$$

in which $A \in \mathbb{R}^{2\times2}, B \in \mathbb{R}^{2\times1}$, $x \in \mathbb{R}^2$ and $n = 1,2 \ldots, N$.

The quadratic stability for the hybrid system (4.1) can be achieved if there exists the feedback control law of $u[n] = A[n] + B[n] \times K[n]$ in which $A[n] + B[n] \times K[n]$ share the CQLF $x^T Mx$, $M = M^T > 0$ in which $K \in R^{1\times2}$ and $M \in R^{2\times2}$. The aim of this algorithm is to find the values of $K[n]$ and M that fulfil the mentioned criteria.

Step 1:
Given a non-singular matrix

$$C = \begin{bmatrix} C[1] & C[2] \\ C[3] & C[4] \end{bmatrix},$$

in which there exists an equal matrix $M > 0$ where $C^T MC$ also has the same characteristics if and only if the quadratic function

$$c_3 c_4 x^2 + (c_2 c_3 + c_1 c_4)x + c_1 c_2 > 0 \tag{4.2}$$

has a positive solution $x > 0$. The search for the CQLF is done within a Brunovsky coordinate. All the n subsystems need to be changed into the Brunovsky canonical form $C[n]$ in which $n = 1,2,..,N$.

$$AB[n] = A[n] \times B[n]$$

$$ABi = AB[n] - \left(\left(\frac{A[n] \times AB[n]}{B[n]|AB[n]}[2] \times B[n]\right)\right)$$

$$C[n] = \frac{1}{ABi[n]|B[n]}$$

Step 2:

$C[n]$ transforms the subsystem n on different coordinate planes for each subsystem in Brunovsky coordinates. For uniformity, all of the $C[n]$ are changed to subsystem one coordinate planes in the Brunovsky form. The transformation named $T[k]$ in which $k = 1,2, \dots, n-1$ is as follows;

$$T[k] = C[1] \times C[k+1]^{-1}$$

Step 3:

$T[k]$ has one $T^T MT$ characteristic in which it will only occur if the quadratic equation

$$(T[2,1] \cdot T[2,2]) \cdot x^2 + (T[1,2] \cdot T[2,1] + T[1,1] \cdot T[2,2]) \cdot x + T[1,1] \cdot T[1,2] > 0$$

in which

$$T[k] = \begin{bmatrix} T[1,1] & T[1,2] \\ T[2,1] & T[2,2] \end{bmatrix}$$

and matrix $M[1]$

$$M[1] = \begin{bmatrix} m_1 & m_2 \\ m_2 & m_3 \end{bmatrix}$$

is defined as a quadratic Lyapunov function in the Brunovsky form if and only if $m_2 > 0$, in which there is a positive solution $x > 0$. To make it easier to achieve the solution, $T[k]$ will be explained in three categories depending on the value of $Z[k] = T[k][2,1] \cdot T[k][2,2]$ as

$$S_p = Z[k] > 0,$$
$$S_n = Z[k] < 0,$$
$$S_z = Z[k] = 0,$$

in which S_p, S_n dan S_z are defined as

$$PSz[k] = T[m][1,2] \cdot T[m][2,1] + T[m][1,1] \cdot T[m][2,2]$$
$$QSz[k] = T[m][1,1] \cdot T[m][1,2] \quad , k \in S_z \tag{4.3}$$

$$P[k] = \frac{T[k][1,1]}{T[k]2,1]}, Q[k] = \frac{T[k][1,2]}{T[k][2,2]} \quad , k \in S_p \cup S_n. \tag{4.4}$$

For each $k \in S_z$, the linear form is formed as

$$L[k] = PSz[k] \cdot x + QSz[k] < 0 \tag{4.5}$$

and every $k \in S_p$ and $k \in S_n$ quadratic form is formed as

$$R[k] = x^2 + (P[k] + Q[k]) \cdot x + P[k] \cdot Q[k] < 0, \quad k \in S_p$$
$$R[k] = x^2 + (P[k] + Q[k]) \cdot x + P[k] \cdot Q[k] < 0, \quad k \in S_n \tag{4.6}$$

Step 4:

For a small value ϵ, the CQLF in the Brunovsky coordinate planes are

$$M[1] = \begin{bmatrix} 1 & m_2 \\ m_2 & m_2^2 + \epsilon \end{bmatrix} \tag{4.7}$$

if and only if $H[k](1, m_2, m_3) > 0$ or

$$m_3 + (P[k] + Q[k]) \cdot m_2 + P[k] \cdot Q[k] > 0, \quad k \in S_p,$$
$$m_3 + (P[k] + Q[k]) \cdot m_2 + P[k] \cdot Q[k] < 0, \quad k \in S_n, \tag{4.8}$$
$$PSz[k] \cdot m_2 + QSz[k] > 0, k \in S_z.$$

Because $m_3 > m_2^2$, the equation (4.9) can be written as

$$\begin{aligned} &\epsilon + m_2^2 + (P[k] + Q[k]) \cdot m_2 + P[k] \cdot Q[k] > 0, \quad k \in S_p, \\ &\epsilon + m_2^2 + (P[k] + Q[k]) \cdot m_2 + P[k] \cdot Q[k] < 0, \quad k \in S_n, \\ &PSz[k] \cdot m_2 + QSz[k] > 0, k \in S_z. \end{aligned} \tag{4.9}$$

in which $\epsilon > 0$.

Step 5:

The value x and ϵ obtained provides a solution in the search of the CQLF $M[1]$ in 4.7. Then, return $M[k]$ to the respective coordinates in the Brunovsky plane as

$$M[k+1] = T[k]^T \times (M[1] \times T[k])$$

Then return the condition of the subsystems to the original planes in Brunovsky plane through:

$$\tilde{A}[n] = C[n] \times (A[k] \times C[k]^{-1})$$

Step 6:

The feedback controller for the system (3.1) in the Brunovsky form can be obtained from

$$u = k[n]x = \left[\left(-1 - \tilde{A}[n][2,1]\right) \quad \left(\frac{-1 \cdot M[n][2,2] - M[n][1,1]}{M[n][1,2]} - \tilde{A}[n][2,2]\right)\right] x$$

by letting $(\tilde{A}, \tilde{b})$ represent the system (4.1) in the form canonical $\tilde{A}[n] = C[n] \times A[n] \times C^{-1}[n]$, $\tilde{b}[n] = [0 \quad 1]^T$, $n = 1,2, \dots, N$ as

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ \tilde{A}[n][2,1] & \tilde{A}[n][2,2] \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u.$$

Step 7:

Lastly, the feedback controller for the system (3.1) can be transformed into the original planes with

$$K[n] = k[n] \times C[n], \qquad n = 1,2, \dots, N \tag{4.10}$$

Step 8:

$$M[0] = C[1]^T \times (M[1] \times C[1])$$

is defining the CQLF for the system (4.1) with the original planes

Step 9:

Its stability can be proven through the Lyapunov function

$$M[0] \times (A[n] + (B[n] \times K[n])) + ((A[n] + (B[n] \times K[n]))^T \times M[0]) < 0 \tag{4.11}$$

in which this system is quadratically stable.

4.3 HYBRID SYSTEM CONTROLER DESIGN TOOL

Figure 4.1 shows the hybrid system controller design tool developed to prove and to apply the algorithm suggested in Section 4.2. This tool is developed using symbolic computation software called Maple. It is especially designed for second order linear time-invariant systems with single input single output (SISO) with switched systems with arbitrary switching. From Figure 4.1, only three steps are required to guarantee the stability of a particular system.

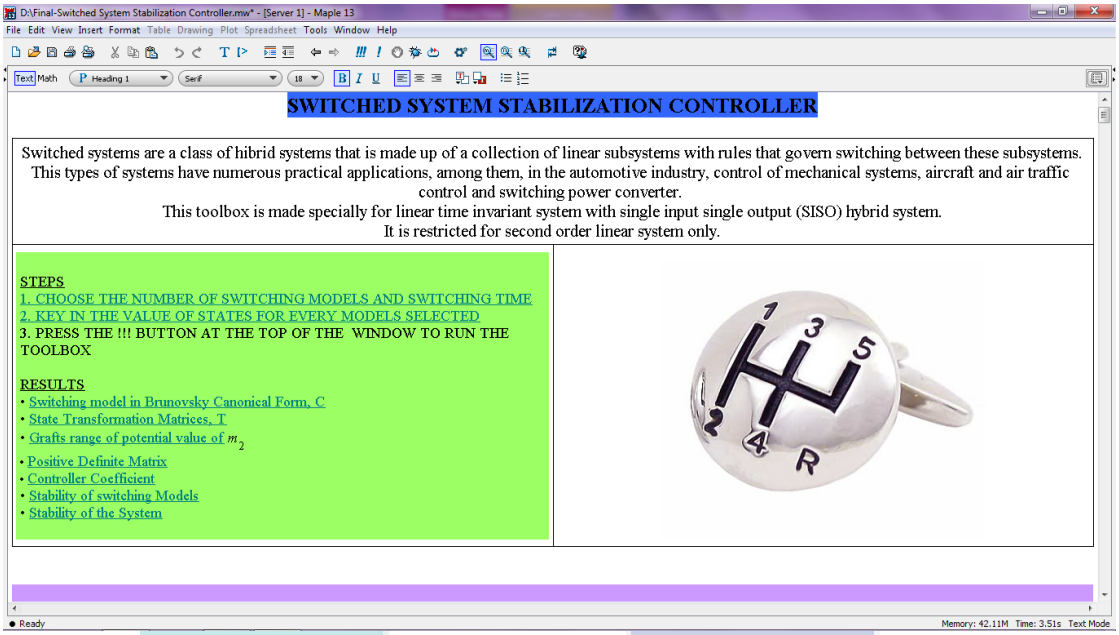


Figure 4.1 Introduction to the hybrid system controller toolbox

Step 1: Choosing the number of subsystems and switching time

Step one is linked to the toolbox like in Figure 4.2. Choose the number of subsystems to be tested and the switching time is entered in seconds.

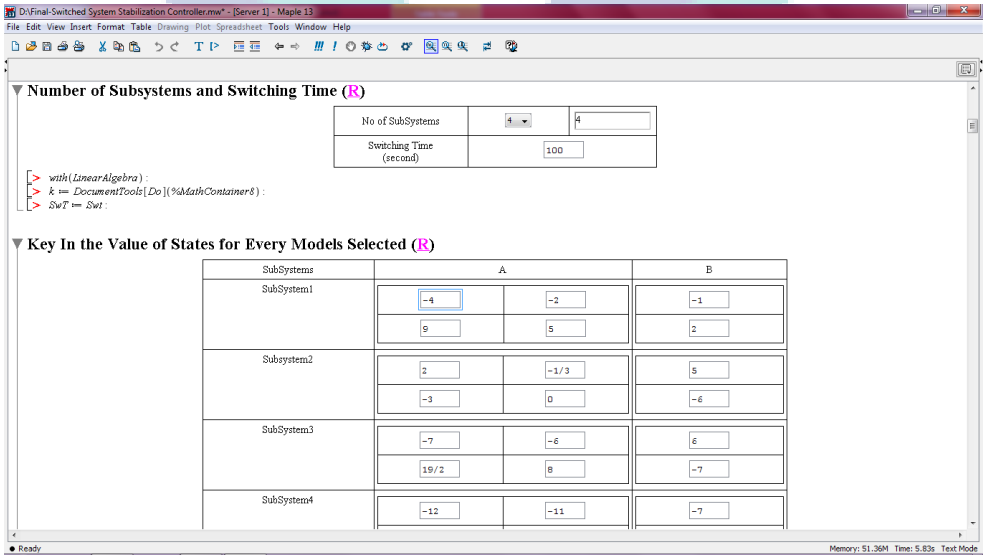


Figure 4.2 The toolbox for entering the required information

Step 2: Key in the value of states for every models selected.

Step two is linked to the state space A and B toolbox from the equation 4.1 for every subsystem. The related values are then entered to their respective boxes as shown in the Figure 4.2 above. The spaces or boxes provided for the subsystems can cater for up to ten pairs of subsystems, thus implying the capability of the toolbox to solve the problem of stability for up to ten subsystems at once; even though the suggested method does not limit the maximum number of subsystems that can be guaranteed in terms of its stability.

Step 3: Clicking on the !!! button

By clicking the !!! button situated on the upper area of the work space, the computing process will be launched to solve the whole of the algorithm provided.

Example 1:

The hybrid system X as in the equation (3.6) is used to test the presentation of the transient response for this algorithm.

Solution:

Step 1:

All the n subsystems in the Brunovsky canonical form $C[n]$;

$$C[1] = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}$$

$$C[2] = \begin{bmatrix} -2 & -\frac{5}{3} \\ 1 & \frac{2}{3} \end{bmatrix}$$

$$C[3] = \begin{bmatrix} \frac{7}{6} & 1 \\ \frac{4}{3} & 1 \end{bmatrix}$$

$$C[4] = \begin{bmatrix} \frac{8}{3} & \frac{7}{3} \\ -\frac{5}{3} & -\frac{4}{3} \end{bmatrix}$$

Step 2:

The transformation matrix, $T[k]$, $k = 1,2,\dots,n-1$ changes the $C[n]$ to subsystem one coordinate planes in the Brunovsky form;

$$T[1] = \begin{bmatrix} 1 & 4 \\ -1 & -1 \end{bmatrix}$$

$$T[2] = \begin{bmatrix} -4 & 5 \\ 2 & -1 \end{bmatrix}$$

$$T[3] = \begin{bmatrix} -3 & -6 \\ 1 & 1 \end{bmatrix}$$

Step 3:

Categories the $T[k]$ depend on the value of $Z[k] = T[k][2,1] \cdot T[k][2,2]$ which are;

$Z[1] = 1,\quad S_p \quad$ Therefore, $\quad P[1] = -1,\quad Q[1] = -4$

$Z[2] = -2,\quad S_n \quad$ Therefore, $\quad P[2] = -2,\quad Q[2] = -5$

$Z[3] = 1,\quad S_p \quad$ Therefore, $\quad P[3] = -3,\quad Q[3] = -6$

Step 4:

The two sets of LI are formed to find the value of x and ϵ.

$$\text{LI 1}\quad \begin{matrix} x^2 - 5x + 4 > 0, \\ x^2 - 7x + 10 < 0, \\ x^2 - 9x + 18 > 0. \end{matrix} \quad \rightarrow \quad x = 3.49999977257838112$$

$$\text{L1 2}\quad \begin{matrix} \epsilon - 1.25000045 > 0, \\ \epsilon - 2.25 < 0, \\ \epsilon - 1.24999955 > 0, \end{matrix} \quad \rightarrow \quad \epsilon = 1.750000225$$

Step 5:

Return $M[k]$ and $\tilde{A}[n]$ to the respective coordinates in the Brunovsky plane;

$$M[1] = \begin{bmatrix} 1 & 3.49999977257838112 \\ 3.49999977257838112 & 13.99999864 \end{bmatrix}$$

$$M[2] = \begin{bmatrix} 7.99999909700000078 & 0.499999777000000034 \\ 0.49999977800000117 & 2.00000045800000015 \end{bmatrix}$$

$$M[3] = \begin{bmatrix} 15.9999981959999982 & 0.99999953200000036 \\ 0.99999954000000146 & 4.00000091499999932 \end{bmatrix}$$

$$M[4] = \begin{bmatrix} 2.00000000100000008 & 3.49999977257838112 \\ 0.500000682000000252 & 8.00000136200000078 \end{bmatrix}$$

and

$$\tilde{A}[1] = \begin{bmatrix} 0 & 1 \\ 2 & 1 \end{bmatrix}$$

$$\tilde{A}[2] = \begin{bmatrix} 0 & 1 \\ 1 & 2 \end{bmatrix}$$

$$\tilde{A}[3] = \begin{bmatrix} 0 & 1 \\ -1 & 1 \end{bmatrix}$$

$$\tilde{A}[4] = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Step 6:

Find the feedback controller, $k[n]$ for the subsystems in the Brunovsky form;

$$k[1] = [-3 \quad -5.285714175]$$

$$k[1] = [-2 \quad -22.00000803]$$

$$k[1] = [0 \quad -21.00000848]$$

$$k[1] = [-2 \quad -19.99997548]$$

Step 7:

The feedback controller, $K[n]$ of the hybrid system X as in Table 4.1.

Table 4.1 The feedback controller from the Method H using Maple

Subsystem	**Feedback Controller, K_i**	
1	[-11. 2857141799999994	-8. 28571417499999896]
2	[-18. 0000080300000000	-11.3333386899999997]
3	[-28.0000113100000014	-21.0000084800000018]
4	[27.999959140000001422	21.9999673000000016]

Step 8:

The Lyapunov matrix shared by the subsystems is

$$M_0 = \begin{bmatrix} 31.9999977359999974 & 26.4999979560000014 \\ 26.499997962999998421 & 21.9999981829999988 \end{bmatrix}$$

Step 9:

The eigenvalues for all subsystems are;

$$E[1] = \begin{bmatrix} -428.983635258895220 \\ -0.016317641104848235 1 \end{bmatrix}$$

$$E[2] = \begin{bmatrix} -95.9270938607990616 \\ -0.0729708920094178 10 \end{bmatrix}$$

$$E[3] = \begin{bmatrix} -428.485854364229340 \\ -0.0144483357706803872 \end{bmatrix}$$

$$E[4] = \begin{bmatrix} -1223.99271425076222 \\ -0.0005718949237632386 94 \end{bmatrix}$$

The negative solution of eigenvalues in Lyapunov function (4.12) are defining CQLF for a switched system X. Therefore, the switched system X is quadratically stable.

4.4 THE TRANSIENT RESPONSE TEST

Maple is an example of a symbolic computation-based software or known as computer algebra system (CAS) which is capable of solving simple and complicated mathematical problems through the use of the packages provided. This software uses a hardware floating-point environment in which it is a processor used to handle complicated mathematical equations that include decimal numbers which may cause congestion in the computing process while the calculations are being performed. Hence, the use of this processor can reduce the computational demanding and also increase the efficiency of calculation.

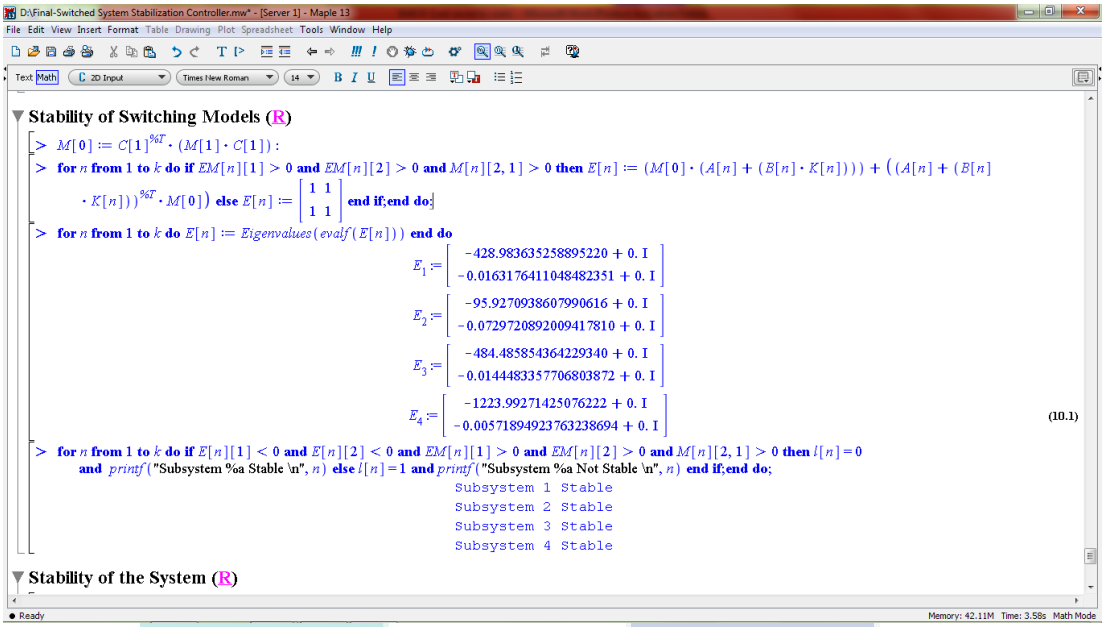


Figure 4.3 The eigenvalues for the switched system X using Maple

Figure 4.4 shows the location of the pole-zero for every subsystem in the hybrid system X without feedback controller verses with feedback controllers that guarantee stability within the switched system with arbitrary switching. For the ease of analysis, the switching sequence is executed in an orderly manner; from subsystem one to subsystem two to subsystem three and subsequently. From Figure 4.4, the location of zero shows that there are no location changes occurring when the hybrid system is tested with the feedback controller from the Method H. Subsystem one and subsystem three are on the right side of the real plane while the rest is on the left side of the plane. Subsystem one and three are non-minimum phase which means that the transient response will follow the calculated derived conditions but on the opposite side.

The location of the pole for a subsystem refers to the level of stability for that particular subsystem. The left side of Figure 4.4 shows the pole locations for each subsystem that are mostly on the right side of the real plane σ. This illustrates that all of the subsystems are individually unstable, which makes the switched system also unstable. However, following the application of the algorithm for finding the feedback controller towards the hybrid system X, each of the subsystems have clearly become stable with the change of the pole locations for each of the subsystem to the left side

of the real plane σ. Due to the fact that the location of the poles are not the same on the left side of the plane, hence the performance of every subsystem are deference. However, considering that the location of one of the poles for each subsystem is seen to be in very close proximity with the imaginary axis $j\omega$, the stability will easily convert into instability if there are any external disturbances. Furthermore, the location for the second pole for subsystem one which is much more close to the imaginary axis $j\omega$ compared to the other subsystems, hence subsystem one will achieve the steady-state much quicker compared to other subsystems as demonstrated in Figure 4.5. Yet, the tool is only considered the stability of the system. The pole placement for the system performance will not be discussed in this thesis.

Figure 4.6 shows transient response for switched system X. However, the steady-state response is varied for every subsystem. Given the lengthy switching time interval, which is 500 seconds as per Figure 4.6 (d), the value of the steady-state is as detailed in Table 4.2.

Figure 4.6 shows the non-minimum phase which due to the one positive value zero in subsystem one and three. Referred to Hag and Bernstein (2007), this situation named as initial undershoot because of there is an odd zero in the right side of real plane. At the beginning of the motion of the subsystem, it will move to the opposite direction from the original direction before it back to its original orientation. This situation will initiate the errors growth in the system which limiting the performance of the tested switched system. More, the loop gain will increase as the error growth, it move the pole towards zero and destabilize the system. Hence, the gain margin is limited when the system is in non-minimum phase which limiting the robustness of the switched system. To overcome this phase, another controller need to be designed.

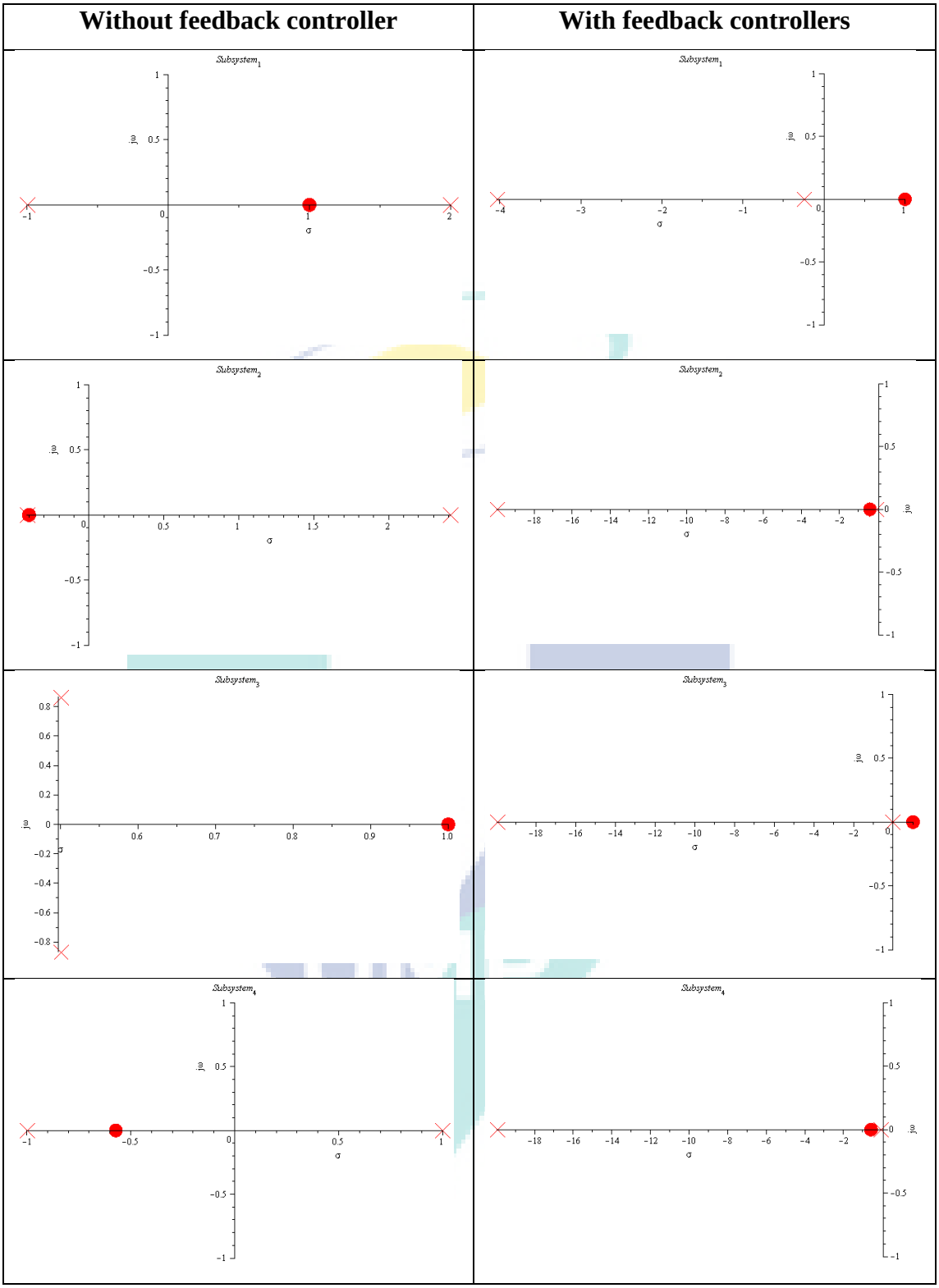


Figure 4.4 The location of the pole-zero for the switched system *X* using the Method *H*

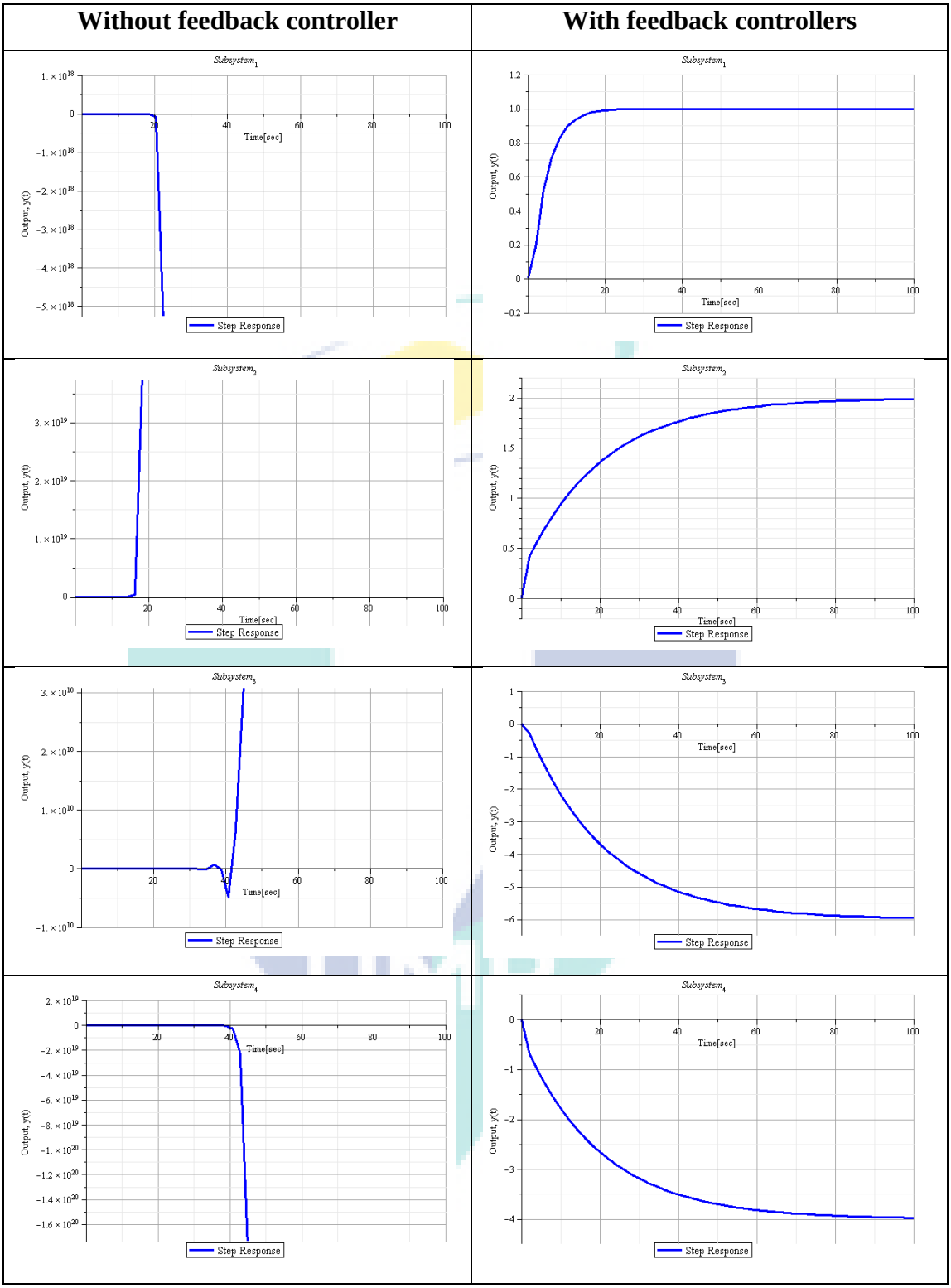


Figure 4.5 Graph of the transient response for the switched system *X* using the Method *H*

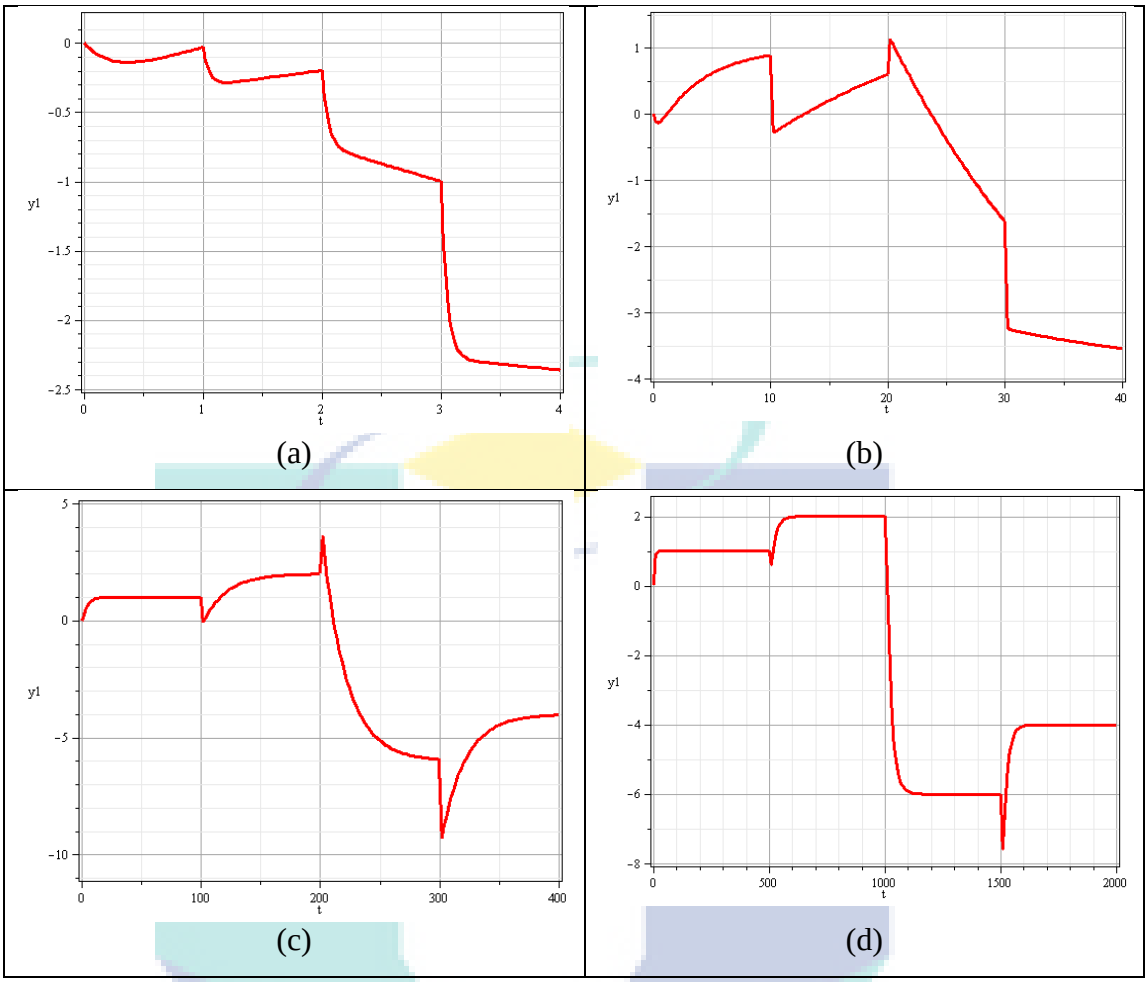


Figure 4.6 Graph of the transient response for the switched system X for the switching interval occurring every (a) 1 second, (b) 10 seconds, (c) 100 seconds and (d) 500 seconds.

Table 4.2 The steady-state response for the switched system X using the Method H

Subsystem	**Steady-state Response**
1	1
2	2
3	-6
4	-4

CHAPTER V

COMPUTING CAPABILITY TEST FOR METHOD *H*

5.1 INTRODUCTION

In the study of switched systems for a hybrid system, LMIs are used to connect the relationship between one subsystem to another. Through the LMI, an algorithm can be constructed; in which it may possibly guarantee the stability of a switched system. Conventionally, the LMI will be analysed and solved by using a numerical computation-based software like Matlab. In Haris et al (2007) or Method H, they have suggested an algorithm constructed on two sets of LI which they insist as capable of guaranteeing stability of a switched system. Then, Haris and Rogers (2008) have developed the "Matlab toolbox" using numerical computation systems which is the Matlab and the LMI Solver to solve the set of LI to acquire the feedback controller for the hybrid system X.

Subsequently, Chapter IV of this thesis has detailed the proof in which two sets of LI from the Method H can also be solved using symbolic computation-based programs like Maple. To further reinforce the Method H, the two sets of LI will be tested in terms of its computational demanding using two numerical computation-based programs like the cvx and Yalmip; as used to test the Method M previously in Chapter III. Besides that, a brief discussion on the use of the numerical computation-based program the LMI Solver from Haris and Rogers (2008) will also be presented.

To begin with, the use of the symbolic computation-based programs Maple to solve the two sets of LI from the Method H will be discussed in detail. Then, the use

of the LMI Solver to solve the two sets of LI will be explained before the test of computing capabilities is performed using the cvx and Yalmip.

5.2 SYMBOLIC COMPUTATION

Even though this symbolic computation-based software does not provide a specific solver to solve an LMI or LI, the use of the optimisation package is the most suitable package to achieve the solution for the two sets of LI that will be tested. In this package, the computing system will identify the command input through three forms which are the algebraic form, the operational form and the matrix form. If the command input is algebraic and operational form, it will be changed into the matrix form to reduce the unwanted side commands.

The first stage is to find the possible value of x to fulfil the first LI (LI-1):

$$x^2 + (P[k] + Q[k]) \cdot x + P[k] \cdot Q[k] > 0, \quad k \in S_p,$$
$$x^2 + (P[k] + Q[k]) \cdot x + P[k] \cdot Q[k] < 0, \quad k \in S_n,$$
$$PSz[k] \cdot x + QSz[k] > 0, k \in S_z.$$

In Haris and Rogers (2008), the minimum value and the maximum value of x which fulfils LI-1 as in Figure 5.1 is pursued. Then, any value of x in that interval can be used as the value of m_2 as per the verification of Lemma 2 in 4.2. However, to make it easy to find the value of x in this toolbox, the use of Non-Linear Programming (NLP) has been utilised. For any system tested, the NLP is used to find the minimum value of y within the LI range to obtain the x value that will fulfil LI-1 as in Figure 5.1.

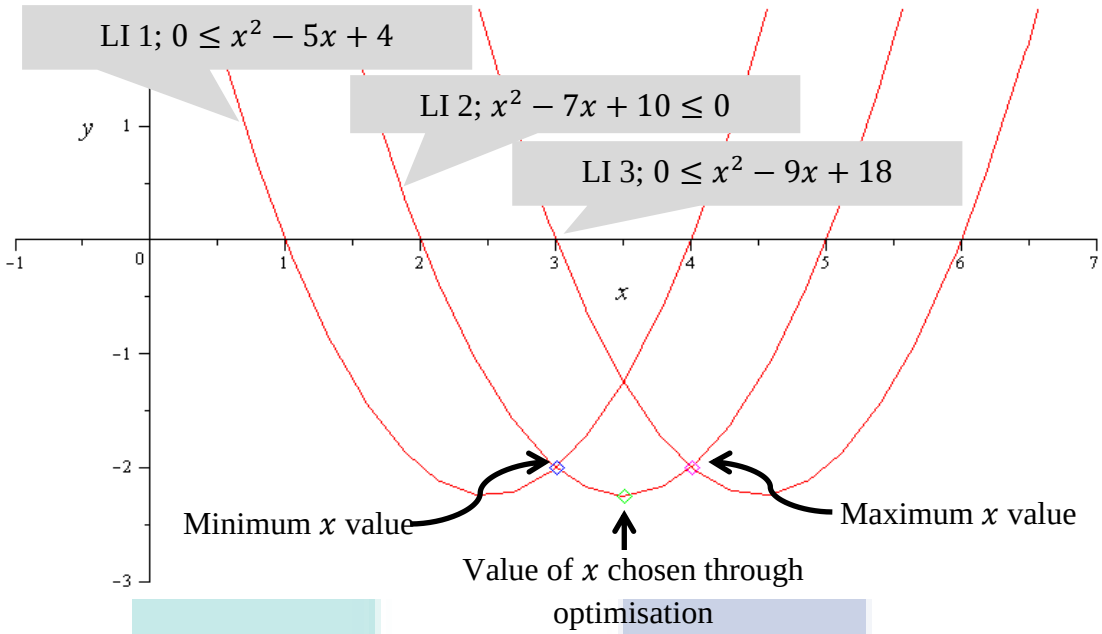


Figure 5.1 The plotted x range

The x value obtained from the solution of LI-1 is used to find the possible value of ϵ from the second LI (LI-2) as in (4.9).

$$\epsilon + x^2 + (P[k] + Q[k]) \cdot x + P[k] \cdot Q[k] > 0, \quad k \in S_p,$$
$$\epsilon + x^2 + (P[k] + Q[k]) \cdot x + P[k] \cdot Q[k] < 0, \quad k \in S_n,$$
$$PSz[k] \cdot x + QSz[k] > 0, k \in S_z.$$

The utilisation of Linear Programming (LP) from algebraic commands is carried out to find the minimum value and maximum value of ϵ for the LI-2. Then, a special command is made to find the middle value for the ϵ interval that is obtained as shown in Figure 5.2. This special command is different to Haris and Rogers (2008) who both allow users to determine themselves the value of ϵ that they wish to insert within the obtained ϵ range. According to Haris et al. (2007), if the x value is obtained from LI-1, mathematically there will be a solution for the value of ϵ for LI-2.

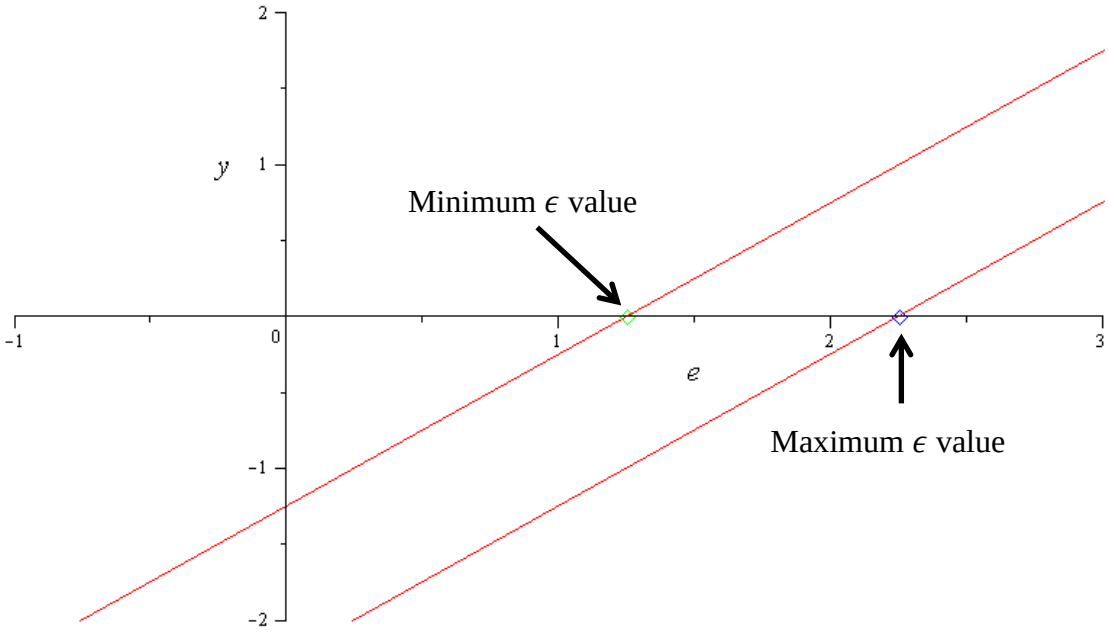


Figure 5.2 The plotted ϵ range

The feedback controller, Lyapunov matrix, eigenvalues, location of the pole-zero and the transient response of the switched system *X* can be referred to in Section 4.4. The solution for the Method *H* can be achieved with the use of 43.24 Megabytes of computing powers for 3.22 seconds as shown in Figure 5.3.

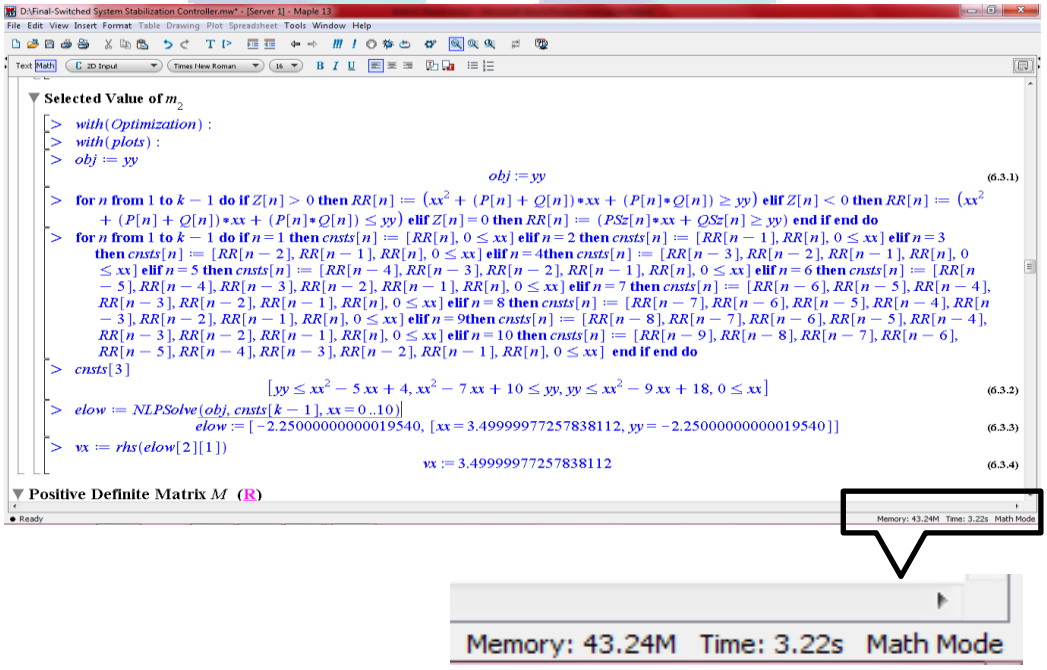


Figure 5.3 The solution for Method *H* using Maple

5.3 NUMERICAL COMPUTATION

5.3.1 LMI Solvers

As discussed in Section 3.3.2, the LMI Solver can be used to find the solution for the `feasp` function for an LMI in the linear form of $A(x) < B(x)$. To solve LI-1 which is in the quadratic form, the LMI Solver logically is unsuitable to be used. However, Haris and Rogers (2008) have converted the quadratic form of LI-1 as follows

$$x^2 + (P[k] + Q[k]) \cdot x + P[k] \cdot Q[k] > 0, \quad k \in S_p,$$
$$x^2 + (P[k] + Q[k]) \cdot x + P[k] \cdot Q[k] < 0, \quad k \in S_n,$$

into a more simple program which is by comparing the values of x obtained in stages which is as follows;

```
for k=1:N-2;
    if B(k,1)>0 and B(k+1,1)<0;
        f3=find ((p(k,1)+(q(k,1)-p(k+1,1)-
q(k+1,1))*x+(p(k,1)*q(k,1))-(p(k+1,1)*q(k+1,1))>0);
    elseif B(k+1,1)>0 and B(k,1)<0;
        f3=find ((p(k,1)+(q(k,1)-p(k+1,1)-
q(k+1,1))*x+(p(k,1)*q(k,1))-(p(k+1,1)*q(k+1,1))>0);
    end
end

for k=1:N-1
    if B(k,1)==0;
        f4=find(p(k,1)*x+q(k,1)>0;
    end
end
```

The solution for LI-2 using the `feasp` function is as follows;

```
setlmis([]);
e=lmivar(2,[1 1]);

for k=1:N-1
    if B(k,1)>0
        lmiterm([-1 1 1 e],1,1);
        lmiterm([-1 1 1
0],(X^2)+((p(k,1)+(q(k,1))*X)+(p(k,1)*q(k,1)));
    elseif B(k,1)<0
        lmiterm([2 1 1 e],1,1);
        lmiterm([2 1 1
0],(X^2)+((p(k,1)+(q(k,1))*X)+(p(k,1)*q(k,1)));
    end
end

lmiterm([-4 1 1 e],1,1);

gete=getlmis;

[tmin, vector]=feasp(gete);
e=dec2mat(gete,vector,e);
```

From this LMI Solvers, the feedback controller acquired for every subsystem using the Method *H* is detailed in the Table 5.1 below.

Table 5.1 The feedback controller from the Method *H* using the LMI Solver

Subsystem	**Feedback Controller, K_i**
1	[-11.29 -8.29]
2	[-18 -11.33]
3	[-28 -21]
4	[28 22]

Source: Haris and Rogers (2008)

The Lyapunov matrix shared by all of the subsystems is

$$M_0 = \begin{bmatrix} 32.0 & 26.5 \\ 26.5 & 22.0 \end{bmatrix}.$$

and the negative solution of eigenvalues in Lyapunov function is defining the CQLF for a switched system X shown in Table 5.5. Figure 5.4 is the transient response for the hybrid system X using the LMI Solver for the switching time intervals every 1 second, 10 seconds, 100 seconds and 500 seconds.

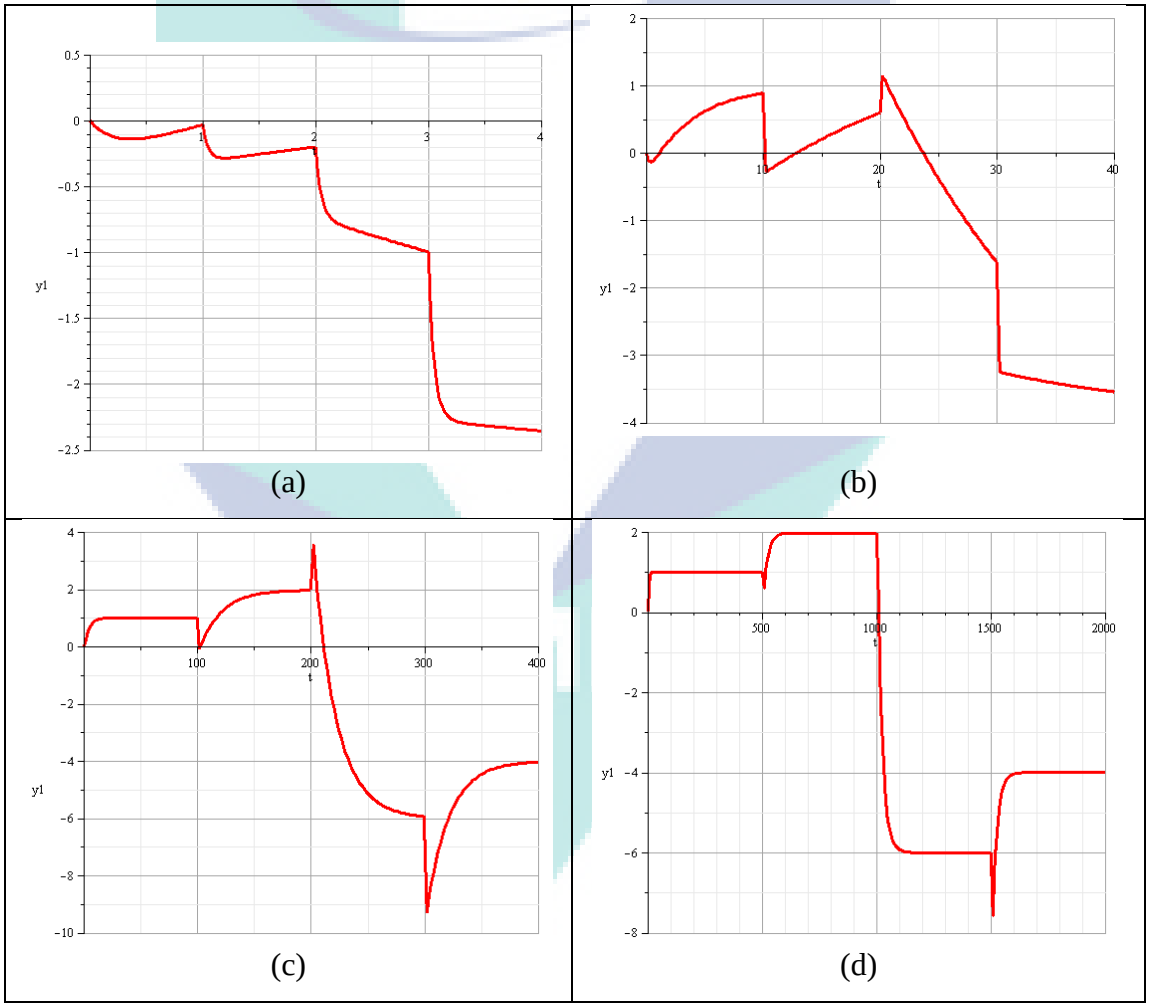


Figure 5.4 The transient response for the switching time intervals for every (a) 1 second, (b) 10 seconds, (c) 100 seconds, and (d) 500 seconds using the LMI Solver for Method *H*

5.3.2 cvx

The definition of cvx can be referred to in Section 3.3.1. Referring to the DCP set by Grant and Boyd (2011), for every convex equation, each of the convexity scalar must start from the power $p \geq 1$ and $p \neq 3, 5, 7, 9, \ldots$ Meanwhile, each of the concavity scalar must be up the power of $p \in (0,1)$ only. This means that only linear concave equations are allowed. Therefore, LI-1 which is in the concave quadratic form needs to be modified into a linear form as follows;

$$\big(x + p(k,1)\big)' \times p(k,1) \times \big(x + q(k,1)\big) \geq 0 \tag{4.7}$$

without changing or interfering with the validity of LI-1. The cvx programming to solve LI-1 is as follows;

```
cvx_begin
variable Xn
variable Yn
minimize Xn
subject to
for k=1:N-1
    if B(k,1)==0;
(p(k,1)*Xn+q(k,1))>=Yn;
    elseif B(k,:)>=0;
(Xn+p(k,1))'*p(k,1)*(Xn+q(k,1))>=Yn;
    else B(k,1)<=0;
(Xn^2+(p(k,1)+q(k,1))*Xn+p(k,1)*q(k,1))<=Yn;
    end
end
cvx_end

cvx_begin
variable Xx
variable Yx
```

```
maximize Xx
subject to
for k=1:N-1
    if B(k,1)==0;
(p(k,1)*Xx+q(k,1))>=Yx;
    elseif B(k,:)>=0;
(Xx+p(k,1))'*p(k,1)*(Xx+q(k,1))>=Yx;
    else B(k,1)<=0;
(Xx^2+(p(k,1)+q(k,1))*Xx+p(k,1)*q(k,1))<=Yx;
    end
end
cvx_end

M2=(Xx+Xn)/2
```

LI-2 which is linear convex and concave in nature can be solved as follows;

```
cvx_begin
variable en
variable Yen
minimize en
subject to
for k=1:N-1
    if B(k,1)==0;
en+p(k,1)*M2+q(k,1)>=Yen;
    elseif B(k,:)>=0;
en+M2^2+(p(k,1)+q(k,1))*M2+p(k,1)*q(k,1)>=Yen;
    else B(k,1)<=0;
en+M2^2+(p(k,1)+q(k,1))*M2+p(k,1)*q(k,1)<=Yen;
    end
end
en>=0;
en<=10;
```

```
Yen==0;
cvx_end

cvx_begin
variable ex
variable Yex
maximize ex
subject to
for k=1:N-1
    if B(k,1)==0;
ex+p(k,1)*M2+q(k,1)>=Yex;
    elseif B(k,:)>=0;
ex+M2^2+(p(k,1)+q(k,1))*M2+p(k,1)*q(k,1)>=Yex;
    else B(k,1)<=0;
ex+M2^2+(p(k,1)+q(k,1))*M2+p(k,1)*q(k,1)<=Yex;
    end
end
ex>=0;
ex<=10;
Yex==0;
cvx_end
e=(ex+en)/2
```

The full solution for the Method *H* can be referred to in the Appendix B.1. The feedback controller obtained when using the cvx to find the solution for the two sets of LI is as follows;

Table 5.2 The feedback controller from the Method *H* using the cvx

Subsystem	**Feedback Controller, K_i**
1	[-11.402913968856573 -8.402913968856574]
2	[-10.687448453596975 -6.458298969064620]
3	[-38.636527223007754 -28.977395417255817]
4	[36.863772685840530 29.091018148672386]

The matrix Lyapunov defined CQLF shared by all of the subsystems is

$$M_0 = \begin{bmatrix} 33.368437298790745 & 27.754400857224852 \\ 27.754400857224852 & 23.140364415658958 \end{bmatrix}$$

and negative solution of eigenvalues of Lyapunov function guaranteed the stability of switched system X shown in Figure 5.5. Figure 5.5 is the transient response for the switched system X for the switching intervals for every 100 seconds, 200 seconds, 300 seconds, and 400 seconds.

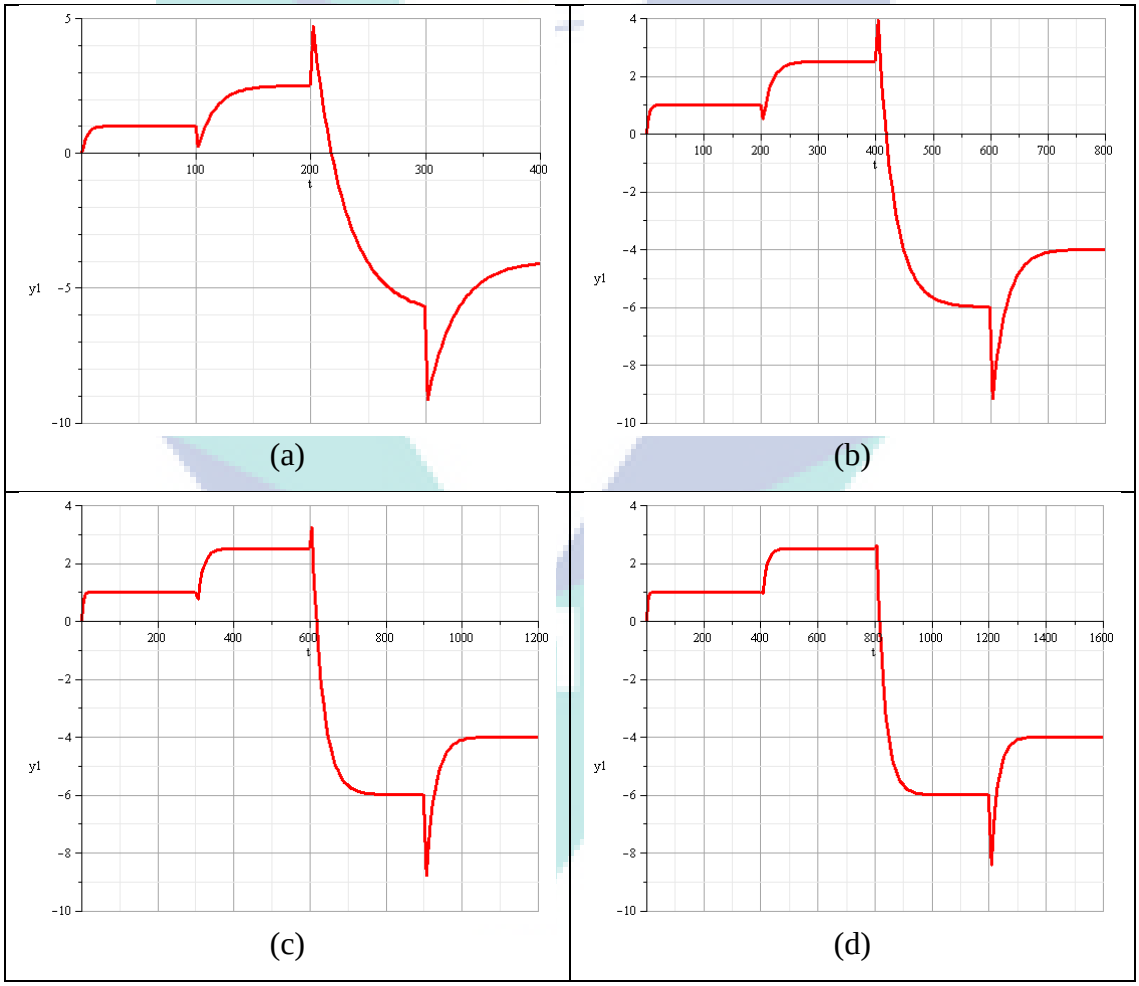


Figure 5.5 The transient response for the switching time intervals for every (a) 100 seconds, (b) 200 seconds, (c) 300 seconds, and (d) 400 seconds using the cvx for the Method *H*

5.3.3 Yalmip

The definition of Yalmip can be referred to in Section 3.3.3. LI-1 is a combination of programs in the quadratic form and in the linear form, depending on the combination of subsystems tested. Versions of LMI and LI solvers for solving this LI which are free and also commercialised can be referred to in Löfberg (2011). A type of semidefinite programming solver called SeDuMi-1.3 is installed to achieve a more accurate solution. SeDuMi-1.3 is a short term for "Self-Dual-Minimization" is the additional to Matlab tools. It is used to solve linear optimization, quadratic and semidefinite problems. Based on Yalmip developer, SeDuMi is the mostly used external solver for the Yalmip user.

Attention should also be given to the type of input and output variable data for the LI. The output for LI-1 has been introduced as an integer type of data, while the input variable is a decimal type; hence leading to the need of another type of solver which is "bnb". Nonetheless, the Yalmip has actually been equipped with an internal solver installed called the bnb for solving problems of linear mixed integer data, quadratic, second order cone, semidefinite types, and also geometric programming problems. The bnb can also be used in conjunction with an external solver. Therefore, LI-1 can successfully be solved as follows;

```
X=intvar(1,1);
F=[X>0];
F=[F,0<X<10];
F=[F,(((X^2+(p(1,1)+q(1,1))*X+p(1,1)*q(1,1))<=0)),(((X^2+
(p(2,1)+q(2,1))*X+p(2,1)*q(2,1))<=0)),(((X^2+(p(3,1)+q(3,
1))*X+p(3,1)*q(3,1))<=0))];
MN=solvesdp(F,X)
Xn=double(X)
checkset(F)

X=intvar(1,1);
G=[X>0];
```

```
G=[G,0<X<10];
G=[G,(((X^2+(p(1,1)+q(1,1))*X+p(1,1)*q(1,1))<=0)),(((X^2+
(p(2,1)+q(2,1))*X+p(2,1)*q(2,1))<=0)),(((X^2+(p(3,1)+q(3,
1))*X+p(3,1)*q(3,1))<=0))];
MX=solvesdp(G,-X)
Xx=double(X)
checkset(G)

M2=(Xx+Xn)/2
```

Then, LINPROG has been suggested to solve LI-2. The bnb is also used for the input data of the decimal type and also the output data of integer type. LI-2 is solved as follows;

```
eN=intvar(1,1);
H=[eN+M2^2+(p(1,1)+q(1,1))*M2+p(1,1)*q(1,1)<0,eN+M2^2+(p(
2,1)+q(2,1))*M2+p(2,1)*q(2,1)<0,eN+M2^2+(p(3,1)+q(3,1))*M
2+p(3,1)*q(3,1)<0];
obj=eN;
EN=solvesdp(H,-obj)
en=double(eN)
checkset(H)

eX=intvar(1,1);
J=[eX+M2^2+(p(1,1)+q(1,1))*M2+p(1,1)*q(1,1)>0,eX+M2^2+(p(
2,1)+q(2,1))*M2+p(2,1)*q(2,1)<0,eX+M2^2+(p(3,1)+q(3,1))*M
2+p(3,1)*q(3,1)>0];
obje=eX;
EX=solvesdp(J,-obje)
ex=double(eX)
checkset(J)

e=(ex+en)/2
```

The full solution for the Method *H* using the Yalmip can be referred to in the Appendix B.3. The feedback controller for the hybrid system *X* is detailed in Table 5.3 below:

Table 5.3 The feedback controller from the Method *H* using the Yalmip

Subsystem	**Feedback Controller, K_i**
1	[-11.214285714285715 -8.214285714285715]
2	[-36.000000000002139 -23.333333333334668]
3	[-18.000000000000043 -13.500000000000030]
4	[58.000000000014730 46.000000000011724]

The Lyapunov matrix defined CQLF is

$$M_0 = \begin{bmatrix} 31.75 & 26.25 \\ 26.25 & 21.75 \end{bmatrix}$$

and the negative solution of eigenvalues from Lyapunov function (4.12) shown in Table 5.5 guaranteeing that the hybrid system *X* is also stable when the Yalmip program is used. The transient response for the switched system *H* following the use of the feedback controllers from Table 5.3 is shown in Figure 5.6.

5.4 DISCUSSION

In Chapter III, the Method *M* has been put to test in terms of its level of consumption of computational demands or powers in solving a set of LMI for the switched system for the hybrid system X. Three types of numerical computation-based programs which are “cvx”, LMI Solver and Yalmip have been used alongside one type of symbolic computation-based program which is Maple. From the tests done, only Yalmip are capable of solving the set of LMI and all together the feedback controller, in which the Lyapunov matrix and negative solution of eigenvalue from Lyapunov function are guaranteeing the stability of the hybrid system X.

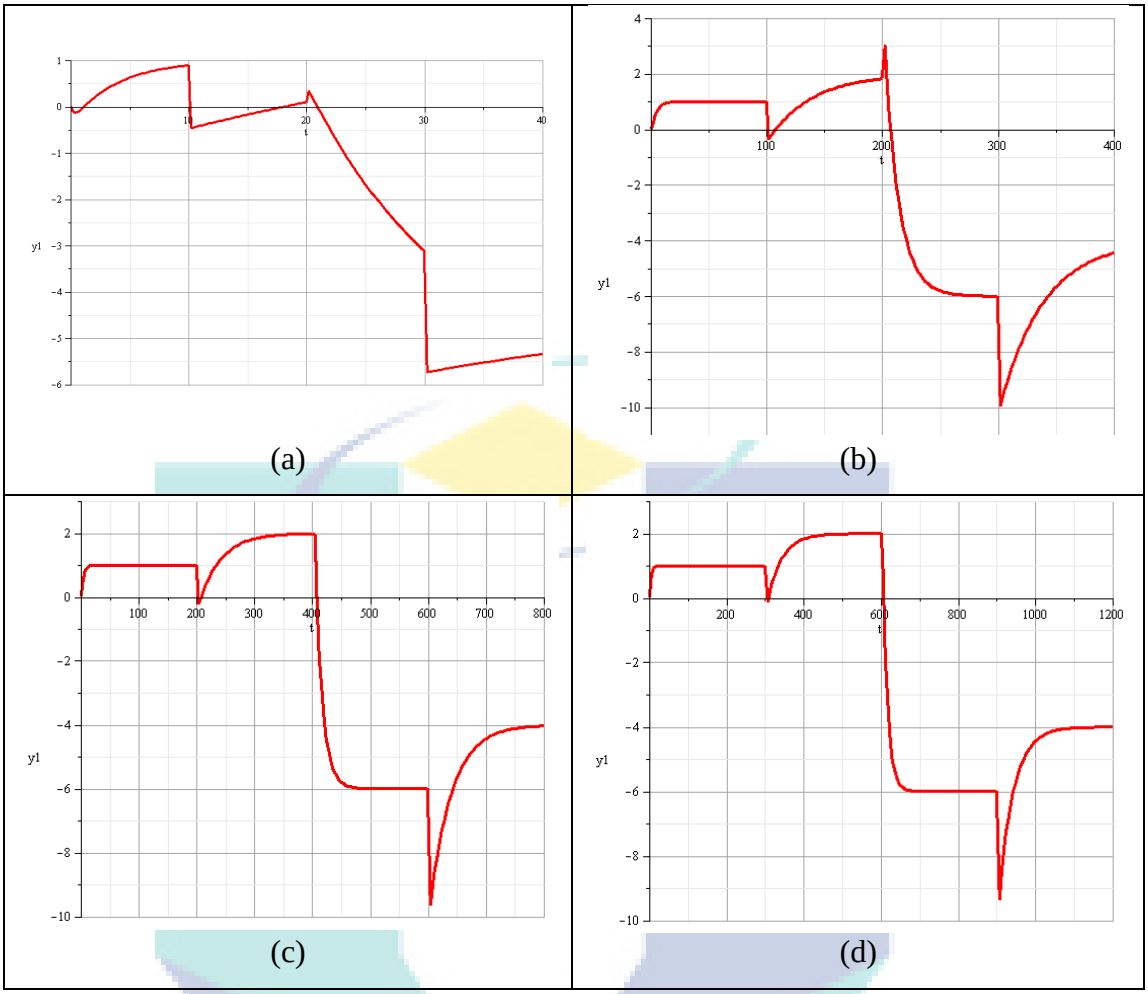


Figure 5.6 The transient response for the switching time intervals for every (a) 10 seconds, (b) 100 seconds, (c) 200 seconds, and (d) 300 seconds using the Yalmip for the Method *H*

From Chapter IV, the suggested Method *H* from Haris et al. (2007) has been developed using the symbolic computation-based software Maple. Meanwhile, in Chapter V, two sets of LI from the Method *H* are tested for the same purposes using the same type of program. The results of the tests show that the two sets of LI from the Method *H* can be solved to obtain the feedback controller for the switched system *X* by using all programs. The Lyapunov matrix and eigenvalues of switched system *X* using Method *M* and Method *H* are in Table 5.4 and 5.5 respectively.

Table 5.4 Lyapunov matrix and eigen values for the switched system X using Method M for different types of computing programs

Type of Computing Program	**Lyapunov Matrix and Eigenvalue of Each Subsystems**	**Stability of System**
cvx	Not suitable	
LMI Solver	$M_0 = 10^3 \begin{bmatrix} 6.474263251043679 & -7.858962346491398 \\ -7.603631937065424 & 9.273498841521807 \end{bmatrix}$ Subsystem 1= $10^4 \begin{bmatrix} -0.035624850184896 \\ -1.768638755871335 \end{bmatrix}$ Subsystem 2= $10^4 \begin{bmatrix} -0.001832713694448 \\ -1.457155689081830 \end{bmatrix}$ Subsystem 3= $10^4 \begin{bmatrix} 0.001175834210283 \\ 1.231520367147503 \end{bmatrix}$ Subsystem 4= $10^3 \begin{bmatrix} -0.010496481177116 \\ -6.088499374290110 \end{bmatrix}$	Not guaranteed
Yalmip	$M_0 = 10^2 \begin{bmatrix} 3.795014078598726 & -4.578381133091899 \\ -4.578381133091899 & 5.539646919034535 \end{bmatrix}$ Subsystem 1= $10^2 \begin{bmatrix} -1.788622818453521 \\ -0.242187329217442 \end{bmatrix}$ Subsystem 2=$10^2 \begin{bmatrix} -1.788622818516506 \\ -0.002964983988222 \end{bmatrix}$ Subsystem 3= $10^2 \begin{bmatrix} -1.788622818384629 \\ -0.002499399799944 \end{bmatrix}$ Subsystem 4= $10^2 \begin{bmatrix} -1.788622818594694 \\ -0.002816654155023 \end{bmatrix}$	Guaranteed
Maple	Needs high levels of computing powers	

Table 5.5(a) Lyapunov matrix and eigenvalues for the switched system X using Method H for different types of computing programs

Type of Computing Program	**Lyapunov Matrix and Eigenvalue of Each Subsystems**	**Stability of System**
cvx	$M_0 = \begin{bmatrix} 33.368437298790745 & 27.754400857224852 \\ 27.754400857224852 & 23.140364415658958 \end{bmatrix}$ Subsystem 1= $10^3 \begin{bmatrix} -0.463245880674346 \\ -0.000015983150277 \end{bmatrix}$ Subsystem 2= $10^3 \begin{bmatrix} -0.057269582773575 \\ -0.000129285532872 \end{bmatrix}$ Subsystem 3= $10^3 \begin{bmatrix} -0.624172878035841 \\ -0.000011862304157 \end{bmatrix}$ Subsystem 4= $10^3 \begin{bmatrix} -1.518376033878618 \\ -0.000004876347073 \end{bmatrix}$	Guaranteed
LMI Solver	$M_0 = \begin{bmatrix} 32.000000000000000 & 26.500000000000000 \\ 26.500000000000000 & 22.000000000000000 \end{bmatrix}$ Subsystem 1= $10^3 \begin{bmatrix} -0.429313695430464 \\ -0.000016304569536 \end{bmatrix}$ Subsystem 2= $10^3 \begin{bmatrix} -0.095922961802304 \\ -0.000073704864362 \end{bmatrix}$ Subsystem 3= $10^3 \begin{bmatrix} -0.484485551684719 \\ -0.000014448315281 \end{bmatrix}$ Subsystem 4= $10^3 \begin{bmatrix} -1.223994281019031 \\ -0.000005718980970 \end{bmatrix}$	Guaranteed

Table 5.5(b) Lyapunov matrix and eigenvalues for the switched system X using Method H for different types of computing programs

Type of Computing Program	**Lyapunov Matrix dan Eigenvalue of Each Subsystems**	**Stability of System**
Yalmip	$M_0 = \begin{bmatrix} 31.750000000000000 & 26.250000000000000 \\ 26.250000000000000 & 21.750000000000000 \end{bmatrix}$ Subsystem 1= $10^3 \begin{bmatrix} -0.417771352362701 \\ -0.000014361923014 \end{bmatrix}$ Subsystem 2= $10^3 \begin{bmatrix} -0.172965310963419 \\ -0.000034689036586 \end{bmatrix}$ Subsystem 3= $10^3 \begin{bmatrix} -0.297479830565360 \\ -0.000020169434643 \end{bmatrix}$ Subsystem 4= $10^3 \begin{bmatrix} -2.452997554013643 \\ -0.000002445986950 \end{bmatrix}$	Guaranteed
Maple	$M_0 = \begin{bmatrix} 31.9999977359999974 & 26.4999979560000014 \\ 26.4999979629999984 & 21.9999981829999988 \end{bmatrix}$ Subsystem 1= $10^3 \begin{bmatrix} -0.428983635258895220 \\ -0.000016317641104848235 \end{bmatrix}$ Subsystem 2= $10^3 \begin{bmatrix} -0.0959270938607990616 \\ -0.000072972089200941781 0 \end{bmatrix}$ Subsystem 3= $10^3 \begin{bmatrix} -0.484485854364229340 \\ -0.000014448335770680 3872 \end{bmatrix}$ Subsystem 4= $10^3 \begin{bmatrix} -1.22399271425076222 \\ -0.000005718949237632 38694 \end{bmatrix}$	Guaranteed

Referred to Table 5.4 and 5.5, there are a huge difference of results while using Method M and Method H. The problem of cramming computational process obviously happen when there are massive difference of results between programs of numerical computational-based. The LMI required high level of computational power to be solved using symbolic computational-based program. However, in their paper, Montagner et al. (2006) have shown two more suggested Lemmas. The stability of the closed loop for a switched system with the performance of the transient response being always positive can be ensured if the suggested LMI takes into account the pole location for every subsystem. An LMI capable of reducing the levels of external disturbances is also suggested to reinforce the switched system from any disturbances.

Contradictory happen when using Method *H* to guarantee stability of switched system *X* as in Table 5.5. Lyapunov matrix and negative solution of eigenvalue for each subsystem are uniform even slighty difference on the values. However, all subsystems are stables individually as shown in Figure 5.7 guarantee stability of the switched system *X* when all eigenvalues have negative solution for all subsystems. The eigenvalues until fifteen points are required to guarantee stability of the whole system.

The decimal numbers for every calculation are important in the research of stability of switched system. Any changes of value may transform the negative solution of eigenvalue to positive solution which not guarantee stability of switched system. Therefore, the selection of the type of numerical data is important for the data calculation. Different data types will provide different memory capacity for the analysis. In this research, data type "long" is used instead of "short" which is the default data type of numeric computational-based, Matlab. The memory capacity of "long" is bigger with the capacity of eight bytes with fifteen decimal number compared to "short" with two bytes and only four decimal number. This bigger capacity is used to prevent any lost information such as decimal number due to the rounding off error that brings differentiation in stability of a switched system. Table 5.6 is an example of rounding off errors. Lyapunov matrix using cvx is rounded until four points and one point proved instability of switched system *X* when the eigenvalues have mixed positive and negative solution for every subsystem.

In this research, external solver type semidefinite named SeDuMi-1.3 has been chosen to solve LI-1 by using Yalmip. Nevertheless, there are more external solver can be suggested to be used such as LMILAB, SDPT3, SDPA, DSDP and many more. Yet, it is difficult to identify which external solver is suitable to be used to solve the problems because of there are no proper references that indicated the external solver with less rounding off errors. Moreover, the final solutions usually are slightly difference when there is a new update of SeDuMi was installed.

Table 5.6 Rounding Off the Decimal of Lyapunov Matrix and The eigenvalues for Switched System X using Method H for cvx

cvx	**Lyapunov Matrix**	**Eigenvalue for Each Subsystems**	**Stability of System**
Rounded Until Four Decimal	$M_0 = \begin{bmatrix} 33.3684 & 27.7544 \\ 27.7544 & 23.1403 \end{bmatrix}$	Subsystem 1= $10^3 \begin{bmatrix} -0.4632 \\ 0 \end{bmatrix}$ Subsystem 2= $10^3 \begin{bmatrix} -0.0573 \\ -0.0001 \end{bmatrix}$ Subsystem 3= $10^3 \begin{bmatrix} -0.6242 \\ 0 \end{bmatrix}$ Subsystem 4= $10^3 \begin{bmatrix} -1.5183 \\ 0 \end{bmatrix}$	Not Guaranteed
Rounded Until One Decimal	$M_0 = \begin{bmatrix} 33.4 & 27.8 \\ 27.8 & 23.1 \end{bmatrix}$	Subsystem 1= $10^3 \begin{bmatrix} -0.4625 \\ 0 \end{bmatrix}$ Subsystem 2= $10^3 \begin{bmatrix} -0.0618 \\ 0.0006 \end{bmatrix}$ Subsystem 3= $10^3 \begin{bmatrix} -0.6242 \\ 0.0011 \end{bmatrix}$ Subsystem 4= $10^3 \begin{bmatrix} -1.5183 \\ 0.0006 \end{bmatrix}$	Not Guaranteed

Table 5.7 and 5.8 shown the feedback controllers and Lyapunov matrix and eigenvalues of switched system X when LMILAB has been used to solve LI-1 and LINPROG for LI-2. The values are difference compared to using SeDuMi-1.3 and while the matrix Lyapunov is rounded until fifteen and four decimal number. As a result, the solutions are not certain for the stability of switched system. The full program of using LMILAB and LINPROG in Yalmip can be referred in Appendix B.3.

Table 5.7 The Feedback Controller from Method H using Yalmip (LMILAB and LINPROG)

Subsystem	**Feedback Controller, K_i**
1	[-11.500000000000000 -8.500000000000000]
2	[-8.999999999999968 -5.333333333333286]
3	[-16.000000000000028 -12.000000000000020]
4	[-16.999999999999442 -13.999999999999545]

Table 5.8 Rounding Off the Decimal of Lyapunov Matrix and The eigenvalues for Switched System X using Method H for Yalmip

Yalmip	**Lyapunov Matrix and Eigenvalue for Each Subsystems**	**Stability of System**
Rounded until Fifteen Decimal	$M_0 = \begin{bmatrix} 37 & 31 \\ 31 & 26 \end{bmatrix}$ Subsystem 1= $10^2 \begin{bmatrix} -5.339925092582186 \\ -0.0000749074178130 \end{bmatrix}$ Subsystem 2= $10^2 \begin{bmatrix} -0.298660687473191 \\ -0.001339312526813 \end{bmatrix}$ Subsystem 3= $10^2 \begin{bmatrix} -1.409716254956561 \\ -0.000283745043440 \end{bmatrix}$ Subsystem 4= $10^2 \begin{bmatrix} 0.000082305920629 \\ 4.859917694079162 \end{bmatrix}$	Not Guaranteed
Rounded until Four Decimal	$M_0 = \begin{bmatrix} 37 & 31 \\ 31 & 26 \end{bmatrix}$ Subsystem 1= $\begin{bmatrix} -533.9925 \\ -0.0075 \end{bmatrix}$ Subsystem 2= $\begin{bmatrix} -29.8661 \\ -0.1339 \end{bmatrix}$ Subsystem 3= $\begin{bmatrix} -140.9716 \\ -0.0284 \end{bmatrix}$ Subsystem 4= $\begin{bmatrix} 0.0082 \\ 485.9918 \end{bmatrix}$	Not Guaranteed

Compared to the usage of symbolic computational-based software or computer algebra system (CAS), any variables in algebraic equation will remain in variable form in algebraic equation. Every step of calculations will be simplified in algebraic form. For the example, the input of symbolic calculation; $y = 3x - x + 1$, the output of symbolic calculation will be $y = 2x + 1$. The value will be reveal when there is any value is introduced to any variable. Furthermore, symbolic computation is dealing with an accurate value such as fraction number ($\frac{1}{3}$) and π instead of 0.333 and 3.142 in numerical computational-based software. The usage of this kind of software is sufficient to stability research where every point is significant. Computer algebra processor is used hardware floating-point environment to handle the decimal which might cause the cramming in computerized process while doing calculation. This processor will reduce computational taxing and together will increase the efficiency of calculation. Therefore, the transition from utilizing numerical computational-based software to symbolic computational-based software is needed to gain the exact solution to guarantee stability of a switched system.

As dealing with numerical computational-based software to test the Method H, the same test has to be done using symbolic computational-based software such as Mathematica, Mathcad, MuPAD and others which capable to solve the algorithm and LMIs. The software must be able to solved a switched system consist of many subsystems and higher order system. Nevertheless, the accurate solution can be defined using any computational software if the algorithm consists of simple equations such as LIs.

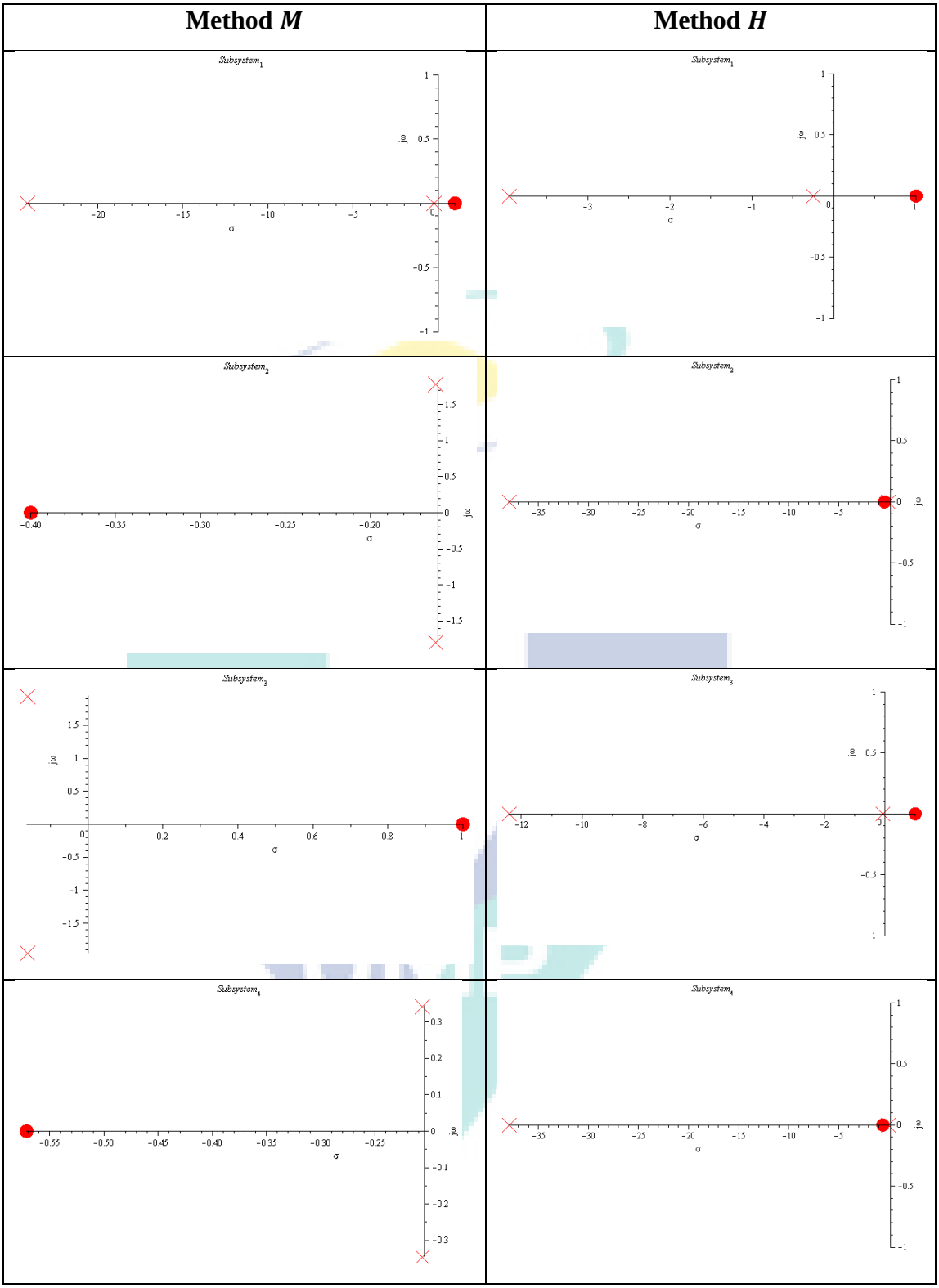


Figure 5.7 Locations of the pole-zero for the Method *M* and Method *H* with feedback controllers using the Yalmip

5.5 CASE STUDY

Three case study as in Table 5.9, 5.10 and 5.11 were calculated using hybrid system controller design tool. The solutions proven the stability of the switched systems.

Table 5.9 Case Study 2

Hybrid System

$$A_1 = \begin{bmatrix} 15 & 22 \\ 35 & 40 \end{bmatrix}, \quad B_1 = \begin{bmatrix} 12 \\ 23 \end{bmatrix}$$

$$A_2 = \begin{bmatrix} 11 & 31 \\ 38 & 42 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 12 \\ 15 \end{bmatrix}$$

$$A_3 = \begin{bmatrix} 10 & 33 \\ 51 & 64 \end{bmatrix}, \quad B_3 = \begin{bmatrix} 14 \\ 20 \end{bmatrix}$$

$$A_4 = \begin{bmatrix} 17 & 63 \\ 52 & 64 \end{bmatrix}, \quad B_4 = \begin{bmatrix} 15 \\ 16 \end{bmatrix}$$

Lyapunov Matrix and Eigenvalue of Each Subsystems

$$M_0 = \begin{bmatrix} 0.000464229889801324504 & -0.000291720080331125845 \\ -0.000291720080331125845 & 0.000324246917178807986 \end{bmatrix}$$

Subsystem 1= $\begin{bmatrix} -0.00312928039976713511 \\ -0.000083628806532865104 6 \end{bmatrix}$

Subsystem 2= $\begin{bmatrix} -0.0322332318480303970 \\ -0.000008118890069603516 66 \end{bmatrix}$

Subsystem 3= $\begin{bmatrix} -0.0388645506106123362 \\ -0.000006733592387658110 53 \end{bmatrix}$

Subsystem 4= $\begin{bmatrix} -0.0655187824931445412 3 \\ -0.000004014524754601526 45 \end{bmatrix}$

Stability of System

Guaranteed

Table 5.10 Case Study 3

Hybrid System (Montagner et al. (2006)) $A_1 = \begin{bmatrix} 0 & 1 \\ -3 & -0.2 \end{bmatrix}$, $B_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ $A_2 = \begin{bmatrix} -0.2 & -3 \\ 1 & 0 \end{bmatrix}$, $B_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$
Lyapunov Matrix and Eigenvalue of Each Subsystems $M_0 = \begin{bmatrix} 1 & 5 \\ 5 & 30 \end{bmatrix}$ Subsystem 1= $\begin{bmatrix} -0.0537712132886554174 \\ -371.946228786711345 \end{bmatrix}$ Subsystem 2= $\begin{bmatrix} -1.90581788928322382 \\ -10.4941821107167784 \end{bmatrix}$
Stability of System Guaranteed

Table 5.11 Case Study 4

Hybrid System

$$A_1 = \begin{bmatrix} -0.1 & 1 \\ -10 & -0.1 \end{bmatrix}, \quad B_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$A_2 = \begin{bmatrix} -0.1 & 10 \\ -1 & -0.1 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Lyapunov Matrix and Eigenvalue of Each Subsystems

$$M_0 = \begin{bmatrix} 30 & -0.2000000000000000010 \\ -0.199999999999999954 & 0.00300000000000000006 \end{bmatrix}$$

Subsystem 1= $\begin{bmatrix} -362.015647537692473 \\ -0.000552462307528232139 \end{bmatrix}$

Subsystem 2= $\begin{bmatrix} -562.556206979969261 \\ -0.000355520030742670358 \end{bmatrix}$

Stability of System

Guaranteed

CHAPTER VI

CONCLUSION

6.1 SUMMARY

The beginning of this thesis started with the discussion of the tendency of the control systems research community towards the extracting of the behaviours within switched systems such as controllability, observability, and stability. This was succeeded by detailed discussions of methods that can be used to achieve stability within switched systems especially for linear time-invariant systems with arbitrary switching sequences. From the study, five popular methods can be concluded which are the introduction to multicontrollers, the Hurwitz stability, state transformation (algebraically and by the use of softwares), strategies of controlling switching stability, and the use of Lie Algebra. Generally, most methods formulate the problem into a set of Linear Matrix Inequalities or LMI before it is solved using any programs which are numerical computation-based. However, previous studies have focused more on proving the validity of the algorithm without any tendency or efforts to apply and find its solution. Most of the LMIs generated are of higher orders that may give an impact on the process of finding the solution by using numerical computation-based systems; in that making it much more difficult and it would not be wrong to say that it will also be computational demanding in the process.

Therefore, a test is carried out against a set of LMI to obtain the feedback controller that can guarantee the stability of a hybrid system. The results show that, from four of the computing programs used and tested – which include three numerical computation-based programs and one symbolic computation-based program – only

Yalmip was capable to solve the particular LMI. High computational demanding is required to solve a set of LMI by using symbolic computational-based software.

Then, an algorithm created by Haris et al. (2007) was brought forth and discussed; in which this algorithm is said to guarantee the stability of switched systems within a hybrid system with arbitrary switching. This algorithm uses a well-known method of state transformation or more specifically using transformation to the Brunovsky canonical form. The suggested algorithm seeks to prove that it is less computationally demanding. This was done by describing or elaborating the algorithm against a symbolic computation-based system. As far as knowledge goes throughout this study, this is the first time such has been used in the study of stability of switched systems. In this thesis, the symbolic computation-based program "Maple 13" was chosen. The algorithm was constructed as a computer program in the form of the hybrid system controller toolbox. Two sets of Linear Inequalities or LI suggested have been successfully solved with the use of this symbolic computation-based software. Computing tests to solve these LI were also executed by using the three numerical computation-based programs. The Lyapunov matrix defined CQLF and the negative definite of eigenvalues for all subsystems guaranteed stability of the whole system. The issue of rounding off error in numerical computational-based software was discussed in detail. The test revealed the guarantee of stability became uncertain when the rounding off were at different decimal precision. The usage of different external solvers also brought the same conclusion in stability of switched systems.

As a result, a suggestion to shift from using conventional program to computer algebra has been made. This program allows the exact solution for the entire calculation. The processor using hardware floating-point environment reduced the cramming in computing process while doing calculation. This type of computation is sufficient to the research that every decimal is important to find eigenvalues that guarantee stability of a switched system.

The location of the pole-zero and the transient response for the switched system with four subsystems in a hybrid system at different switching time intervals were illustrated as references. Due to the fact that there are some subsystems which

are in the non-minimum phase, the steady state responses for both methods are not maintained to show the accepted response for the real time. Nonetheless, this result is acceptable and reasonable because the suggested methods did not take into account the performance of the system.

6.2 SUGGESTIONS

Suggestions for future studies are to test Method *H* using various symbolic computational-based software to strengthen the suggestion of shifting the programs. Optimizing the performance by a few ways; rectifying the location of the poles so that it will be more towards the left direction from the real plane σ (pole placement), the root locus, the Linear Quadratic Regulator (LQR) and the Linear Quadratic Guassion Control (LQG) to fulfil the constraints of the system performance. The toolbox should also be expanded to subsystems of a higher order so that it can be used with multiple input multiple outputs (MIMO) systems and also discrete time systems. For the higher order system, polynomial methods would be required to replace quadratic functions. Symbolic computation based systems allow for the possibility of using ideas from polynomial ring theory. For example the Quantifier Elimination and Sum of Roots method (Anai et al. 2009) could be used.

REFERENCES

Agrachev, A. A. & Liberzon, D. 1999. Lie-algebraic conditions for exponential stability of switched systems. *Proceedings of the 38th IEEE Conference on Decision and Control*, p. 2679-2684.

Anai, H., S. Hara, M. Kanno, & K. Yokoyama. 2009. Parametric polynomial spectral factorization using the sum of roots and its application to a control design problem. *Journal of Symbolic Computation*, 44 (7):703-725.

Blanchini, F., Miani, S. & Mesquine, F. 2008. A separation principle for linear switching systems and parametrization of all stabilizing controllers. *47th IEEE Conference on Decision and Control*, p. 953-958.

Boyd, S. 2009. EE363 Review Session4: Linear Matrix Inequalities. Stanford University.

Brayton, R. K. & Tong, C. H. 1979. Stability of dynamical systems: A constructive approach. *IEEE Transaction on Circuit and Systems*, 26(4): 224-234.

Brayton, R. K. & Tong, C. H. 1980. Constructive stability and asymptotic stability of dynamical systems. *IEEE Transaction on Circuit and Systems*, 27(11): 1121-1130.

Brockett, R. W. 1993. Hybrid models for motion control systems. *Essays on Control: Perspectives in the Theory and its Applications*, p. 29-53.

Davrazos, G. & Koussoulas, N. T. 2002. A general methodology for stability analysis of differential petri nets. *Proceedings of the 10th Mediterranean Conference on Control and Automation.*

Davrazos, G. & Koussoulas, N. T. 2007. Modeling and stability analysis of state-switched hybrid systems via Differential Petri Nets. *Simulation Modelling Practice and Theory*, 15(8): 879-893.

Dayawansa, W. P. & Martin, C. F. 1999. A converse Lyapunov theorem for a class of dynamical systems which undergo switching. *IEEE Transactions on Automatic Control*, 44(4): 751-760.

Decarlo, R. A., Branicky, M. S., Pettersson, S. & Lennartson, B. 2000. Perspectives and results on the stability and stabilizability of hybrid systems. *Proceedings of the IEEE*, 88(7): 1069-1082.

Ezzine, J. & Haddad, A. H. 1988. On the controllability and observability of hybrid systems. *Proceedings of the 1988 American Control Conference*, p. 41-46.

Fainshil, L., Margaliot, M. & Chigansky, P. 2009. On the stability of positive linear switched systems under arbitrary switching laws. *IEEE Transactions on Automatic Control*, 54(4): 897-899.

Ge, S. S., Zhendong, S. & Lee, T. H. 2001. Reachability and controllability of switched linear systems. *Proceedings of the 2001 American Control Conference*, p. 1898-1903.

Geng, Z. 2010. Switched stability design on canonical forms. *IEEE International Conference on Information and Automation (ICIA)*, p. 289-293.

Gopal, M. 2003. *Control Systems Principles and Design*. Ed. Ke-2. New Delhi: Mc Graw Hill.

Grant, M. & Boyd, S. 2011. Cvx users' guide for cvx version 1.21. http://cvxr.com/cvx/cvx_usrguide.pdf

Guangming, X., Dazhong, Z. & Long, W. 2002. Controllability of switched linear systems. *IEEE Transactions on Automatic Control*, 47(8): 1401-1405.

Guangming, X. & Long, W. 2002. Necessary and sufficient conditions for controllability of switched linear systems. *Proceedings of the 2002 American Control Conference*, p. 1897-1902.

Guisheng, Z., Derong, L., Imae, J. & Kobayashi, T. 2006. Lie algebraic stability analysis for switched systems with Continuous-Time and Discrete-Time subsystems. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 53(2): 152-156.

Gurvits, L., Shorten, R., & Mason, O. (2007). On the stability of switched positive linear systems. *IEEE Transactions on Automatic Control*, 52(6): 1099-1103.

Haris, S. M. & Rogers, E. 2008. A Matlab toolbox for finding stabilizing controllers for a class of switched systems. *International Conference on Computational Intelligence for Modelling Control & Automation*, p. 238-242.

Haris, S. M., Saad, M. H. M. & Rogers, E. 2007. A method for determining stabilizeability of a class of switched system. *Proceedings of the 7th WSEAS International Conference on Systems Theory and Scientific Computation*, p. 27-32.

Haris, S. M. 2006. Analysis and design of classes of hybrid control systems. Thesis Ph.D. University of Southampton.

Hespanha, J. P. & Morse, A. S. 2002. Switching between stabilizing controllers. *Automatica*, 38(11): 1905-1917.

Hespanha, J. P., Santesso, P. & Stewart, G. 2007. Optimal controller initialization for switching between stabilizing controllers. *46th IEEE Conference on Decision and Control*, p. 5634-5639.

Hag, J. B. & Bernstein, D. S. 2007. Nonminimum-phase zeros - Much to do about nothing - Classical control - Revisited Part II. *Control Systems*, 27(3): 45-57.

Jianhong, W., Xun, L., Yaping, G. & Guangfeng, J. 2008. An LMI optimization approach to Lyapunov stability analysis for linear time-invariant systems. *Control and Decision Conference*, 2008. p. 3044-3048.

Jin, L. & Brown, L. J. 2010. A multiple Lyapunov functions approach for stability of switched systems. *American Control Conference*, p. 3253-3256.

King, C. & Shorten, R. 2004. A singularity test for the existence of common quadratic Lyapunov functions for pairs of stable LTI Systems. *Proceedings of the 2004 American Control Conference*, p. 3881-3884.

Li, Z. G., Wen, C. Y. & Soh, Y. C. 2001. Stabilization of a class of switched systems via designing switching laws. *IEEE Transactions on Automatic Control*. 46(4): 665-670.

Liberzon, D. 1999. ISS and integral-ISS disturbance attenuation with bounded controls. *Proceedings of the 38th IEEE Conference on Decision and Control*, p. 2501-2506.

Liberzon, D., Hespanha, J. P. & Morse, A. S. 1999. Stability of switched systems: A lie-algebraic condition. *Systems & Control Letters*, 37(3): 117-122.

Liberzon, D. & Morse, A. S. 1999. Basic problems in stability and design of switched systems. *Control Systems*, 19(5): 59-70.

Ling, H. & Michel, A. N. 1997. Stability analysis of a general class of hybrid dynamical systems. *Proceedings of the 1997 American Control Conference*, p. 2805-2809.

Löfberg, J. 2011. Yalmip. Http://Users.Isy.Liu.Se/Johanl/Yalmip/Pmwiki.Php?N=Solvers.Solvers.

Lyapunov, A. M. & Fuller, A. T. 1992. The general problem of the stability of motion. Terj. Taylor & Francis.

Lygeros, J. 2004. Hybrid Automata & Executions. *Lecture Notes on Hybrid Systems*. 16.

Mancilla-Aguilar, J. L. & García, R. A. 2000. A converse Lyapunov theorem for nonlinear switched systems. *Systems & Control Letters*, 41(1): 67-71.

Martin, C. F. & Dayawansa, W. P. 1996. On the existence of a Lyapunov function for a family of switching systems. *Proceedings of the 35th IEEE Decision and Control*, p. 1820-1823.

Mason, O. & Shorten, R. 2003. A conjecture on the existence of common quadratic Lyapunov functions for positive linear systems. *Proceedings of the 2003 American Control Conference*, p. 4469-4470.

Mason, P., Sigalotti, M. & Daafouz, J. 2007. On stability analysis of linear discrete-time switched systems using quadratic Lyapunov functions. *46th IEEE Conference on Decision and Control*, p. 5629-5633.

Montagner, V. F., Leite, V. J. S., Oliveira, R. C. L. F. & Peres, P. L. D. 2006. State feedback control of switched linear systems: An LMI approach. *Journal of Computational and Applied Mathematics*, 194(2): 192-206.

Moor, T., Davoren, J. M. & Raisch, J. 2006. Learning by doing: Systematic abstraction refinement for hybrid control synthesis. *Control Theory and Applications, IEE Proceedings*, 153(5): 591-599.

Narendra, K. S. & Shorten, R. 2010. Hurwitz stability of Metzler matrices. *IEEE Transactions on Automatic Control*, 55(6): 1484-1487.

Nesic, D. & Liberzon, D. 2005. A small-gain approach to stability analysis of hybrid systems. *44th IEEE Conference on Decision and Control*, p. 5409-5414.

Qi, F., Guangming, X. & Long, W. 2005. Stability analysis and stabilization synthesis for periodically switched linear systems with uncertaintie. *Proceedings of the 2005 American Control Conference*, p. 30-35.

Stewart, G. E. & Dumont, G. A. 2006. Finite horizon based switching between stabilizing controllers. *American Control Conference*, p. 7.

Sun, Z. 2007. Converse Lyapunov theorem for switched stability of switched linear systems. *Chinese Control Conference*, p. 678-680.

Sun, Z. & Ge, S. S. 2005. Analysis and synthesis of switched linear control systems. *Automatica*, 41(2005): 181-195.

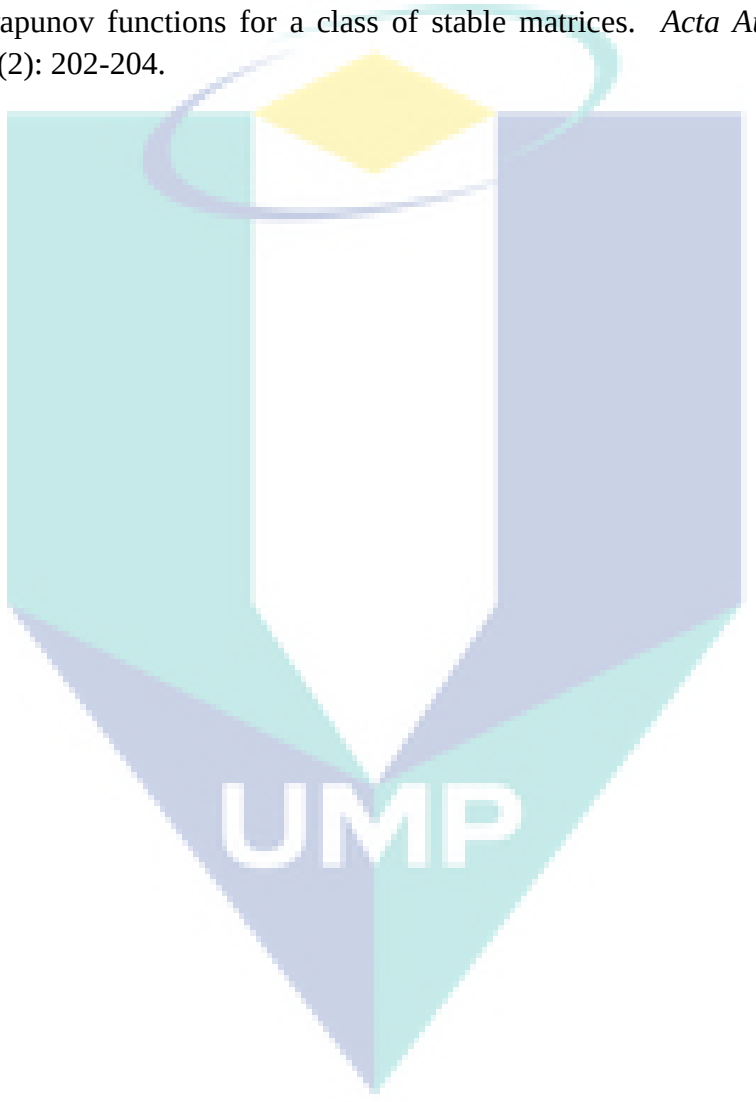
Vu, L. & Liberzon, D. 2006. On invertibility of switched linear systems. *45th IEEE Conference on Decision and Control*, p. 4081-4086.

W. Zhang, Shen, S. Q. & Han, Z. Z. 2008. Sufficient conditions for Hurwitz stability of matrices. *Latin American Applied Research*, 38.

Wenxiang, X., Changyun, W. & Zhengguo, L. 2001. Input-to-State stabilization of switched nonlinear systems. *IEEE Transactions on Automatic Control*, 46(7): 1111-1116.

Yijing, W., Guangming, X. & Long, W. 2003. Reachability and controllability of switched linear systems with state jumps. *IEEE International Conference on Systems, Man and Cybernetics*, p. 672-677.

Zhu, Y. H., Cheng, D. Z. & Qin, H. S. 2007. Constructing common quadratic Lyapunov functions for a class of stable matrices. *Acta Automatica Sinica*, 33(2): 202-204.



APPENDIX A

METHOD *M*

A.1 cvx

INPUT

```
A1=[-4 -2;9 5];
B1=[-1;2];
A2=[2 -1/3;-3 0];
B2=[5;-6];
A3=[-7 -6;19/2 8];
B3=[6;-7];
A4=[-12 -11;13 12];
B4=[-7;8];

cvx_begin
variable W(2,2)
variable Z1(1,2)
variable Z2(1,2)
variable Z3(1,2)
variable Z4(1,2)
subject to
A1'*W+W*A1+B1*Z1+Z1'*B1'<=0;
A2'*W+W*A2+B2*Z2+Z2'*B2'<=0;
A3'*W+W*A3+B3*Z3+Z3'*B3'<=0;
A4'*W+W*A4+B4*Z4+Z4'*B4'<=0;
-W<=0;
cvx_end
```

```
K1=Z1*W^-1
K2=Z2*W^-1
K3=Z3*W^-1
K4=Z4*W^-1
```

OUTPUT

```
Homogeneous problem detected; solution determined
analytically.
Status: Solved
Optimal value (cvx_optval): +0

W =
   All zero sparse: 2-by-2
Z1 =
     0     0
Z2 =
     0     0
Z3 =
     0     0
Z4 =
     0     0

Warning: Matrix is singular to working precision.
K1 =
   NaN   NaN
Warning: Matrix is singular to working precision.
K2 =
   NaN   NaN
Warning: Matrix is singular to working precision.
```

```
K3 =
   NaN   NaN
Warning: Matrix is singular to working precision.
K4 =
   NaN   NaN
```

A.2 LMI Solver

INPUT

```
A1=[-4 -2;9 5];
B1=[-1;2];
A2=[2 -1/3;-3 0];
B2=[5;-6];
A3=[-7 -6;19/2 8];
B3=[6;-7];
A4=[-12 -11;13 12];
B4=[-7;8];

setlmis([]);
  M=lmivar(2,[2 2]);
  Z1=lmivar(2,[1 2]);
  Z2=lmivar(2,[1 2]);
  Z3=lmivar(2,[1 2]);
  Z4=lmivar(2,[1 2]);

lmiterm([1 1 1 M],A1,1,'s');      % LMI #1: A1*M+M*A1'
lmiterm([1 1 1 Z1],B1,1,'s');     % LMI #1: B1*Z1+Z1'*B1'
lmiterm([2 1 1 M],A2,1,'s');      % LMI #2: A2*M+M*A2'
lmiterm([2 1 1 Z2],B2,1,'s');     % LMI #2: B2*Z2+Z2'*B2'
lmiterm([3 1 1 M],A3,1,'s');      % LMI #3: A3*M+M*A3'
lmiterm([3 1 1 Z3],B3,1,'s');     % LMI #3: B3*Z3+Z3'*B3'
```

```
lmiterm([4 1 1 M],A4,1,'s');       % LMI #4: A4*M+M*A4'
lmiterm([4 1 1 Z4],B4,1,'s');      % LMI #4: B4*Z4+Z4'*B4'

lmiterm([5 1 1 M],1,-1);           % LMI #5: -M
N=getlmis;

[tmin,xfeas]=feasp(N);
M=dec2mat(N,xfeas,M)
Z1=dec2mat(N,xfeas,Z1)
Z2=dec2mat(N,xfeas,Z2)
Z3=dec2mat(N,xfeas,Z3)
Z4=dec2mat(N,xfeas,Z4)

K1=Z1*M^-1
K2=Z2*M^-1
K3=Z3*M^-1
K4=Z4*M^-1

eig(M*(A1+(B1*K1))'+(A1+(B1*K1))*M)
eig(M*(A2+(B2*K2))'+(A2+(B2*K2))*M)
eig(M*(A3+(B3*K3))'+(A3+(B3*K3))*M)
eig(M*(A4+(B4*K4))'+(A4+(B4*K4))*M)
```

OUTPUT

```
Solver for LMI feasibility problems L(x) < R(x)
This solver minimizes  t  subject to  L(x) < R(x) + t*I
The best value of t should be negative for feasibility

 Iteration   :    Best value of t so far

     1                         0.184311
```

```
     2                        0.021906
     3                    6.523196e-003
     4                    6.523196e-003
     5                       -0.153688

 Result:  best value of t:    -0.153688
          f-radius saturation:  0.002% of R = 1.00e+009

Z1 =
  1.0e+004 *
  -1.027876889736133   1.161280325314555
Z2 =
  1.0e+003 *
  -3.209985647415382   4.065354270370817
Z3 =
  1.0e+002 *
  -1.479194927675230   0.463021891627658
Z4 =
  1.0e+003 *
   0.935476116686323  -1.237234434558113

M =
  1.0e+003 *
   6.474263251043679  -7.858962346491398
  -7.603631937065424   9.273498841521807

K1 =
 -24.859349377217598 -19.815162604854265
K2 =
   4.049560670684983   3.870243552564773
K3 =
  -3.610516536632684  -3.054792134849636
```

```
K4 =
  -2.593123038571612  -2.330996220943554

ans =
    1.0e+004 *
  -0.035624850184896
  -1.768638755871335

ans =
  1.0e+004 *
  -0.001832713694448
  -1.457155689081830

ans =
  1.0e+004 *
   0.001175834210283
   1.231520367147503

ans =
  1.0e+003 *
  -0.010496481177116
  -6.088499374290110
```

A.3 Yalmip

INPUT

```
A1=[-4 -2;9 5];
B1=[-1;2];
A2=[2 -1/3;-3 0];
B2=[5;-6];
A3=[-7 -6;19/2 8];
```

```
B3=[6;-7];
A4=[-12 -11;13 12];
B4=[-7;8];

M=sdpvar(2,2,'symmetric');
Z1=sdpvar(1,2);
Z2=sdpvar(1,2);
Z3=sdpvar(1,2);
Z4=sdpvar(1,2);

F=[(A1*M+M*A1'+B1*Z1+Z1'*B1'<0),(A2*M+M*A2'+B2*Z2+Z2'*B2'
<0),(A3*M+M*A3'+B3*Z3+Z3'*B3'<0),
(A4*M+M*A4'+B4*Z4+Z4'*B4'<0),(-M<0)]

solvesdp(F)
Z1_feasible=double(Z1)
Z2_feasible=double(Z2)
Z3_feasible=double(Z3)
Z4_feasible=double(Z4)
M_feasible=double(M)

K1=Z1_feasible*inv(M_feasible)
K2=Z2_feasible*inv(M_feasible)
K3=Z3_feasible*inv(M_feasible)
K4=Z4_feasible*inv(M_feasible)

eig(M_feasible*(A1+(B1*K1))'+(A1+(B1*K1))*M_feasible)
eig(M_feasible*(A2+(B2*K2))'+(A2+(B2*K2))*M_feasible)
eig(M_feasible*(A3+(B3*K3))'+(A3+(B3*K3))*M_feasible)
eig(M_feasible*(A4+(B4*K4))'+(A4+(B4*K4))*M_feasible)
```

OUTPUT

```
Linear matrix variable 2x2 (symmetric, real, 3 variables)
Linear matrix variable 1x2 (full, real, 2 variables)
Linear matrix variable 1x2 (full, real, 2 variables)
Linear matrix variable 1x2 (full, real, 2 variables)
Linear matrix variable 1x2 (full, real, 2 variables)
++++++++++++++++++++++++++++++++++++++++++++++++
|   ID|      Constraint|                    Type|
++++++++++++++++++++++++++++++++++++++++++++++++
|   #1|   Numeric value|   Matrix inequality 2x2|
|   #2|   Numeric value|   Matrix inequality 2x2|
|   #3|   Numeric value|   Matrix inequality 2x2|
|   #4|   Numeric value|   Matrix inequality 2x2|
|   #5|   Numeric value|   Matrix inequality 2x2|
++++++++++++++++++++++++++++++++++++++++++++++++

 Solver for LMI feasibility problems L(x) < R(x)
    This solver minimizes  t  subject to  L(x) < R(x) +
t*I
    The best value of t should be negative for
feasibility

 Iteration   :    Best value of t so far

     1                      0.039150
     2                   3.817204e-003
     3                   3.817204e-003
     4                   1.422412e-004
     5                      -0.012444

 Result:  best value of t:    -0.012444
          f-radius saturation:  0.000% of R = 1.00e+009
```

```
ans =
    yalmiptime: 0.780000000000001
    solvertime: 0.046999999999997
          info: 'No problems detected (LMILAB)'
       problem: 0
        dimacs: [NaN NaN NaN NaN NaN NaN]

Z1_feasible =
  1.0e+002 *
  -5.747556834678825   6.383763871175754
Z2_feasible =
  1.0e+002 *
  -1.896710236847474   2.377256886308619
Z3_feasible =
 -21.411264144438952  19.123258524136336
Z4_feasible =
  74.437427287211861 -93.299145566445802

M_feasible =
  1.0e+002 *
   3.795014078598726  -4.578381133091899
  -4.578381133091899   5.539646919034535

K1 =
 -42.498387524630857 -33.971488527868246
K2 =
   6.131744340011608   5.496870094902704
K3 =
  -5.052901199286849  -4.141577120035931
K4 =
  -2.408226482848733  -2.158760354899222
```

```
ans =
  1.0e+002 *
  -1.788622818453521
  -0.242187329217442

ans =
  1.0e+002 *
  -1.788622818516506
  -0.002964983988222

ans =
  1.0e+002 *
  -1.788622818384629
  -0.002499399799944

ans =
  1.0e+002 *
  -1.788622818594694
  -0.002816654155023
```

A.4 Maple

> $with(LinearAlgebra)$:

> $A[1] := \begin{bmatrix} -4 & -2 \\ 9 & 5 \end{bmatrix}$:

> $B[1] := \begin{bmatrix} -1 \\ 2 \end{bmatrix}$:

> $A[2] := \begin{bmatrix} 2 & -\frac{1}{3} \\ -3 & 0 \end{bmatrix}$:

> $B[2] := \begin{bmatrix} 5 \\ -6 \end{bmatrix}$:

> $A[3] := \begin{bmatrix} -7 & -6 \\ \frac{19}{2} & 8 \end{bmatrix}:$

> $B[3] := \begin{bmatrix} 6 \\ -7 \end{bmatrix}:$

> $A[4] := \begin{bmatrix} -12 & -11 \\ 13 & 12 \end{bmatrix}:$

> $B[4] := \begin{bmatrix} -7 \\ 8 \end{bmatrix}:$

> $k := 4:$

> **for** n **from** 1 **to** k **do** $A[n] \cdot W + W \cdot A[n]^{\%T} + B[n] \cdot Z[n] + Z[n]^{\%T} \cdot B[n]^{\%T} < 0$; **end do**

$$\begin{bmatrix} -4 & -2 \\ 9 & 5 \end{bmatrix}.W + W.\begin{bmatrix} -4 & 9 \\ -2 & 5 \end{bmatrix} + \begin{bmatrix} -1 \\ 2 \end{bmatrix}.Z_1 + Z_1.\begin{bmatrix} -1 & 2 \end{bmatrix} < 0$$

$$\begin{bmatrix} 2 & -\frac{1}{3} \\ -3 & 0 \end{bmatrix}.W + W.\begin{bmatrix} 2 & -3 \\ -\frac{1}{3} & 0 \end{bmatrix} + \begin{bmatrix} 5 \\ -6 \end{bmatrix}.Z_2 + Z_2.\begin{bmatrix} 5 & -6 \end{bmatrix} < 0$$

$$\begin{bmatrix} -7 & -6 \\ \frac{19}{2} & 8 \end{bmatrix}.W + W.\begin{bmatrix} -7 & \frac{19}{2} \\ -6 & 8 \end{bmatrix} + \begin{bmatrix} 6 \\ -7 \end{bmatrix}.Z_3 + Z_3.\begin{bmatrix} 6 & -7 \end{bmatrix} < 0$$

$$\begin{bmatrix} -12 & -11 \\ 13 & 12 \end{bmatrix}.W + W.\begin{bmatrix} -12 & 13 \\ -11 & 12 \end{bmatrix} + \begin{bmatrix} -7 \\ 8 \end{bmatrix}.Z_4 + Z_4.\begin{bmatrix} -7 & 8 \end{bmatrix} < 0$$

> $solve((1))$

> $EM = eigenvalues(evalf(M))$

$$EM = eigenvalues(M)$$

> **for** n **from** 1 **to** k **do if** $EM[1] > 0$ **and** $EM[2] > 0$ **then** $K[n] = Z[n] \cdot M^{-1}$ **end if**;**end do**

Error, cannot determine if this expression is true or false: 0 < EM[1] and 0 < EM[2]

APPENDIX B

METHOD *H*

B.1 cvx

INPUT

```
A1=[-4 -2;9 5];
b1=[-1;2];
A2=[2 -1/3;-3 0];
b2=[5;-6];
A3=[-7 -6;19/2 8];
b3=[6;-7];
A4=[-12 -11;13 12];
b4=[-7;8];
A=[A1;A2;A3;A4];
b=[b1;b2;b3;b4];
N=4;

for k=1:N
a(2*k-1:2*k,:)=inv([b(2*k-1:2*k,:) A(2*k-1:2*k,:)*b(2*k-
1:2*k,:)])*A(2*k-1:2*k,:)*A(2*k-1:2*k,:)*b(2*k-1:2*k,:);
D2(2*k-1:2*k,:)=b(2*k-1:2*k,:);
D1(2*k-1:2*k,:)=A(2*k-1:2*k,:)*D2(2*k-1:2*k,:)-
a(2*k,:)*b(2*k-1:2*k,:);
D(2*k-1:2*k,:)=[D1(2*k-1:2*k,:) D2(2*k-1:2*k,:)];
C(2*k-1:2*k,:)=inv(D(2*k-1:2*k,:));
end
```

```
for k=1:(N-1)
T(2*k-1:2*k,:)=C(1:2,:)*inv(C(2*k+1:2*k+2,:));
end

for k=1:N-1;
B(k,:)=T(2*k,1)*T(2*k,2);
end

for k=1:N-1;
if B(k,1)==0;
p(k,1)=(T(2*k-1,2)*T(2*k,1))+(T(2*k-1,1)*T(2*k,2));
q(k,1)=T(2*k-1,1)*T(2*k-1,2);
elseif B(k,:)>0
p(k,1)=T(2*k-1,1)/T(2*k,1);
q(k,1)=T(2*k-1,2)/T(2*k,2);
else B(k,1)<0
p(k,1)=T(2*k-1,1)/T(2*k,1);
q(k,1)=T(2*k-1,2)/T(2*k,2);
end
end

cvx_begin
variable Xn
variable Yn
minimize Xn
subject to
for k=1:N-1
    if B(k,1)==0;
(p(k,1)*Xn+q(k,1))>=Yn;
    elseif B(k,:)>=0;
(Xn+p(k,1))'*p(k,1)*(Xn+q(k,1))>=Yn;
    else B(k,1)<=0;
(Xn^2+(p(k,1)+q(k,1))*Xn+p(k,1)*q(k,1))<=Yn;
```

```
    end
end
cvx_end

cvx_begin
variable Xx
variable Yx
maximize Xx
subject to
for k=1:N-1
    if B(k,1)==0;
(p(k,1)*Xx+q(k,1))>=Yx;
    elseif B(k,:)>=0;
(Xx+p(k,1))'*p(k,1)*(Xx+q(k,1))>=Yx;
    else B(k,1)<=0;
(Xx^2+(p(k,1)+q(k,1))*Xx+p(k,1)*q(k,1))<=Yx;
    end
end
cvx_end

M2=(Xx+Xn)/2;

cvx_begin
variable en
variable Yen
minimize en
subject to
for k=1:N-1
    if B(k,1)==0;
en+p(k,1)*M2+q(k,1)>=Yen;
    elseif B(k,:)>=0;
en+M2^2+(p(k,1)+q(k,1))*M2+p(k,1)*q(k,1)>=Yen;
    else B(k,1)<=0;
```

```
en+M2^2+(p(k,1)+q(k,1))*M2+p(k,1)*q(k,1)<=Yen;
    end
end
en>=0;
en<=10;
Yen==0;
cvx_end

cvx_begin
variable ex
variable Yex
maximize ex
subject to
for k=1:N-1
    if B(k,1)==0;
ex+p(k,1)*M2+q(k,1)>=Yex;
    elseif B(k,:)>=0;
ex+M2^2+(p(k,1)+q(k,1))*M2+p(k,1)*q(k,1)>=Yex;
    else B(k,1)<=0;
ex+M2^2+(p(k,1)+q(k,1))*M2+p(k,1)*q(k,1)<=Yex;
    end
end
ex>=0;
ex<=10;
Yex==0;
cvx_end

e=(ex+en)/2;

M1=[1 M2;M2 (M2^2+e)];

for k=1:N-1
M(2*k-1:2*k,:)=T(2*k-1:2*k,:)'*M1*T(2*k-1:2*k,:);
```

```
end

M=[M1;M];

for k=1:N
Ao(2*k-1:2*k,:)=C(2*k-1:2*k,:)*A(2*k-1:2*k,:)*inv(C(2*k-
1:2*k,:));
end

for k=1:N
Ko(k,:)=[-1-Ao((2*k),1) ((((-1)*M((2*k),2))-M((2*k-
1),1))/M((2*k-1),2))-Ao(2*k,2))];
end

for k=1:N
K(k,:)=Ko(k,:)*C(2*k-1:2*k,:)
end

Mo=C(1:2,:)'*M(1:2,:)*C(1:2,:)

for k=1:N
P(2*k-1:2*k,:)=(Mo(1:2,:)*(A(2*k-1:2*k,:)+(b(2*k-
1:2*k,:)*K(k,:))))+(((A(2*k-1:2*k,:)+(b(2*k-
1:2*k,:)*K(k,:)))')*Mo(1:2,:));
end

for k=1:N
    Z(2*k-1:2*k,:)=eig(P(2*k-1:2*k,:));
end

Z
```

OUTPUT

```
Calling sedumi: 12 variables, 7 equality constraints
------------------------------------------------------------
---
SeDuMi 1.21 by AdvOL, 2005-2008 and Jos F. Sturm, 1998-
2003.
Alg = 2: xz-corrector, Adaptive Step-Differentiation,
theta = 0.250, beta = 0.500
eqs m = 7, order n = 10, dim = 16, blocks = 4
nnz(A) = 17 + 0, nnz(ADA) = 35, nnz(L) = 21
 it :     b*y       gap    delta  rate   t/tP*  t/tD*
feas cg cg  prec
  0 :            5.59E+000 0.000
  1 : -1.36E+000 1.43E+000 0.000 0.2562 0.9000 0.9000
2.42  1  1  1.4E+000
  2 : -1.65E+000 3.72E-001 0.000 0.2593 0.9000 0.9000
1.46  1  1  2.7E-001
  3 : -1.38E+000 9.80E-002 0.000 0.2635 0.9000 0.9000
1.01  1  1  7.5E-002
  4 : -1.18E+000 8.19E-003 0.000 0.0836 0.9900 0.9900
1.02  1  1  8.0E-003
  5 : -1.19E+000 5.41E-004 0.000 0.0661 0.9900 0.9900
1.01  1  1  5.3E-004
  6 : -1.19E+000 1.06E-004 0.000 0.1953 0.5187 0.9000
1.00  1  1  2.1E-004
  7 : -1.19E+000 7.89E-007 0.218 0.0075 0.9990 0.9990
1.00  1  1  1.5E-006
  8 : -1.19E+000 3.25E-008 0.000 0.0412 0.6520 0.9000
1.00  1  1  4.1E-007
  9 : -1.19E+000 7.85E-010 0.000 0.0242 0.9900 0.9900
1.00  1  1  1.0E-008
```

```
iter seconds digits       c*x                b*y
  9      0.1    Inf -1.1861406949e+000 -1.1861406863e+000
|Ax-b| =  1.1e-007, [Ay-c]_+ =  5.5E-011, |x|= 1.5e+001,
|y|= 3.2e+000

Detailed timing (sec)
   Pre           IPM           Post
3.120E-002    1.248E-001    3.120E-002
Max-norms: ||b||=1.800000e+001, ||c|| = 1,
Cholesky |add|=0, |skip| = 0, ||L.L|| = 30.4553.
------------------------------------------------------------
---
Status: Solved
Optimal value (cvx_optval): +2.81386

Calling sedumi: 12 variables, 7 equality constraints
------------------------------------------------------------
---
SeDuMi 1.21 by AdvOL, 2005-2008 and Jos F. Sturm, 1998-
2003.
Alg = 2: xz-corrector, Adaptive Step-Differentiation,
theta = 0.250, beta = 0.500
eqs m = 7, order n = 10, dim = 16, blocks = 4
nnz(A) = 17 + 0, nnz(ADA) = 35, nnz(L) = 21
 it :     b*y       gap    delta  rate   t/tP*  t/tD*
feas cg cg  prec
  0 :            5.59E+000 0.000
  1 : -2.78E+000 1.40E+000 0.000 0.2505 0.9000 0.9000
2.13  1  1  1.5E+000
  2 : -1.32E+000 4.49E-001 0.000 0.3205 0.9000 0.9000
1.30  1  1  5.4E-001
  3 : -6.69E-001 1.24E-001 0.000 0.2750 0.9000 0.9000
1.07  1  1  2.1E-001
```

```
  4 : -4.55E-001 1.18E-002 0.000 0.0956 0.9900 0.9900
1.13  1  1  3.0E-002
  5 : -4.22E-001 3.26E-003 0.000 0.2755 0.9000 0.9000
1.05  1  1  8.4E-003
  6 : -4.14E-001 3.02E-005 0.044 0.0093 0.9900 0.9901
1.01  1  1  9.8E-005
  7 : -4.14E-001 2.22E-007 0.000 0.0073 0.9452 0.9900
1.00  1  1  3.2E-006
  8 : -4.14E-001 5.10E-008 0.000 0.2301 0.9000 0.7733
1.00  1  1  7.4E-007
  9 : -4.14E-001 4.80E-009 0.464 0.0941 0.9900 0.9303
1.00  1  1  6.9E-008
 10 : -4.14E-001 1.80E-010 0.000 0.0375 0.9748 0.9900
1.00  2  2  2.6E-009

iter seconds digits       c*x               b*y
 10      0.1   Inf -4.1421357799e-001 -4.1421357415e-001
|Ax-b| =  2.7e-008, [Ay-c]_+ =  1.4E-010, |x|= 2.1e+001,
|y|= 3.8e+000

Detailed timing (sec)
   Pre          IPM          Post
0.000E+000    1.092E-001    0.000E+000
Max-norms: ||b||=1.800000e+001, ||c|| = 1,
Cholesky |add|=0, |skip| = 0, ||L.L|| = 19.4871.
------------------------------------------------------------
---
Status: Solved
Optimal value (cvx_optval): +4.41421

Calling sedumi: 5 variables, 1 equality constraints
   For improved efficiency, sedumi is solving the dual
problem.
```

```
------------------------------------------------------------
---
SeDuMi 1.21 by AdvOL, 2005-2008 and Jos F. Sturm, 1998-
2003.
Alg = 2: xz-corrector, Adaptive Step-Differentiation,
theta = 0.250, beta = 0.500
eqs m = 1, order n = 6, dim = 6, blocks = 1
nnz(A) = 5 + 0, nnz(ADA) = 1, nnz(L) = 1
 it :     b*y       gap    delta  rate   t/tP*  t/tD*
feas cg cg  prec
  0 :            3.29E+001 0.000
  1 :  6.20E+000 9.45E+000 0.000 0.2871 0.9000 0.9000
1.84  1  1  1.3E+000
  2 :  8.23E+000 1.92E+000 0.000 0.2027 0.9000 0.9000
1.65  1  1  1.9E-001
  3 :  8.52E+000 4.29E-001 0.000 0.2240 0.9000 0.9000
1.29  1  1  3.9E-002
  4 :  8.53E+000 9.15E-003 0.000 0.0213 0.9900 0.9900
1.06  1  1
iter seconds digits       c*x                b*y
  4      0.1   Inf  8.5349314269e+000  8.5349314269e+000
|Ax-b| =  0.0e+000, [Ay-c]_+ =  0.0E+000, |x|= 1.0e+000,
|y|= 8.5e+000

Detailed timing (sec)
   Pre          IPM          Post
6.240E-002    6.240E-002    3.120E-002
Max-norms: ||b||=1, ||c|| = 10,
Cholesky |add|=0, |skip| = 0, ||L.L|| = 1.
------------------------------------------------------------
---
Status: Solved
Optimal value (cvx_optval): +1.46507
```

```
Calling sedumi: 5 variables, 1 equality constraints
   For improved efficiency, sedumi is solving the dual
problem.
------------------------------------------------------------
---
SeDuMi 1.21 by AdvOL, 2005-2008 and Jos F. Sturm, 1998-
2003.
Alg = 2: xz-corrector, Adaptive Step-Differentiation,
theta = 0.250, beta = 0.500
eqs m = 1, order n = 6, dim = 6, blocks = 1
nnz(A) = 5 + 0, nnz(ADA) = 1, nnz(L) = 1
 it :     b*y       gap    delta  rate   t/tP*  t/tD*
feas cg cg  prec
  0 :            3.50E+001 0.000
  1 : -4.81E+000 9.15E+000 0.000 0.2617 0.9000 0.9000
1.59  1  1  2.6E+000
  2 : -7.32E+000 2.18E+000 0.000 0.2388 0.9000 0.9000
1.41  1  1  3.9E-001
  3 : -7.76E+000 1.55E-001 0.000 0.0709 0.9900 0.9900
1.61  1  1  1.8E-002
  4 : -7.76E+000 1.10E-004 0.000 0.0007 0.9999 0.9999
1.02  1  1
iter seconds digits       c*x               b*y
  4      0.0   Inf -7.7630043100e+000 -7.7630043100e+000
|Ax-b| =  0.0e+000, [Ay-c]_+ =  0.0E+000, |x|= 1.0e+000,
|y|= 7.8e+000

Detailed timing (sec)
   Pre          IPM          Post
0.000E+000    1.560E-002    0.000E+000
Max-norms: ||b||=1, ||c|| = 10,
Cholesky |add|=0, |skip| = 0, ||L.L|| = 1.
```

```
------------------------------------------------------------
---
Status: Solved
Optimal value (cvx_optval): +2.237

K =
 -11.402913968856573  -8.402913968856574
 -10.687448453596975  -6.458298969064620
 -38.636527223007754 -28.977395417255817
  36.863772685840530  29.091018148672386

Mo =
  33.368437298790745  27.754400857224852
  27.754400857224852  23.140364415658958

Z =
  1.0e+003 *
  -0.463245880674346
  -0.000015983150277
  -0.057269582773575
  -0.000129285532872
  -0.624172878035841
  -0.000011862304157
  -1.518376033878618
  -0.000004876347073
```

B.2 LMI Solver

INPUT

```
A1=[-4 -2;9 5];
B1=[-1;2];
```

```
A2=[2 -1/3;-3 0];
B2=[5;-6];
A3=[-7 -6;19/2 8];
B3=[6;-7];
A4=[-12 -11;13 12];
B4=[-7;8];

N=4;
k1=[-11.29 -8.29];
k2=[-18 -11.33];
k3=[-28 -21];
k4=[28 22];
K=[k1;k2;k3;k4];

Mo=[32 26.5
    26.5 22]

for k=1:N
P(2*k-1:2*k,:)=(Mo(1:2,:)*(A(2*k-1:2*k,:)+(b(2*k-
1:2*k,:)*K(k,:))))+(((A(2*k-1:2*k,:)+(b(2*k-
1:2*k,:)*K(k,:)))')*Mo(1:2,:));
end
 for k=1:N
    Z(2*k-1:2*k,:)=eig(P(2*k-1:2*k,:));
end

Z
```

OUTPUT

```
Mo =
  32.000000000000000  26.500000000000000
  26.500000000000000  22.000000000000000
```

```
Z =
  1.0e+003 *
  -0.429313695430464
  -0.000016304569536
  -0.095922961802304
  -0.000073704864362
  -0.484485551684719
  -0.000014448315281
  -1.223994281019031
  -0.000005718980970
```

B.3 Yalmip

INPUT

```
A1=[-4 -2;9 5];
b1=[-1;2];
A2=[2 -1/3;-3 0];
b2=[5;-6];
A3=[-7 -6;19/2 8];
b3=[6;-7];
A4=[-12 -11;13 12];
b4=[-7;8];
A=[A1;A2;A3;A4];
b=[b1;b2;b3;b4];
N=4;

for k=1:N
a(2*k-1:2*k,:)=inv([b(2*k-1:2*k,:) A(2*k-1:2*k,:)*b(2*k-
1:2*k,:)])*A(2*k-1:2*k,:)*A(2*k-1:2*k,:)*b(2*k-1:2*k,:);
D2(2*k-1:2*k,:)=b(2*k-1:2*k,:);
```

```
D1(2*k-1:2*k,:)=A(2*k-1:2*k,:)*D2(2*k-1:2*k,:)-
a(2*k,:)*b(2*k-1:2*k,:);
D(2*k-1:2*k,:)=[D1(2*k-1:2*k,:) D2(2*k-1:2*k,:)];
C(2*k-1:2*k,:)=inv(D(2*k-1:2*k,:));
end

for k=1:(N-1)
T(2*k-1:2*k,:)=C(1:2,:)*inv(C(2*k+1:2*k+2,:));
end

for k=1:N-1;
B(k,:)=T(2*k,1)*T(2*k,2);
end

for k=1:N-1;
if B(k,1)==0;
p(k,1)=(T(2*k-1,2)*T(2*k,1))+(T(2*k-1,1)*T(2*k,2));
q(k,1)=T(2*k-1,1)*T(2*k-1,2);
elseif B(k,:)>0
p(k,1)=T(2*k-1,1)/T(2*k,1);
q(k,1)=T(2*k-1,2)/T(2*k,2);
else B(k,1)<0
p(k,1)=T(2*k-1,1)/T(2*k,1);
q(k,1)=T(2*k-1,2)/T(2*k,2);
end
end

X=intvar(1,1);
F=[X>0];
F=[F,0<X<10];
F=[F,(((X^2+(p(1,1)+q(1,1))*X+p(1,1)*q(1,1))<0)),(((X^2+(
p(2,1)+q(2,1))*X+p(2,1)*q(2,1))<0)),(((X^2+(p(3,1)+q(3,1)
)*X+p(3,1)*q(3,1))<0))];
```

```
MN=solvesdp(F,X);
Xn=double(X);
checkset(F);

X=intvar(1,1);
G=[X>0];
G=[G,0<X<10];
G=[G,(((X^2+(p(1,1)+q(1,1))*X+p(1,1)*q(1,1))<0)),(((X^2+(
p(2,1)+q(2,1))*X+p(2,1)*q(2,1))<0)),(((X^2+(p(3,1)+q(3,1)
)*X+p(3,1)*q(3,1))<0))];
MX=solvesdp(G,-X);
Xx=double(X);
checkset(G);
M2=(Xx+Xn)/2;

eN=intvar(1,1);
H=[eN+M2^2+(p(1,1)+q(1,1))*M2+p(1,1)*q(1,1)<0,eN+M2^2+(p(
2,1)+q(2,1))*M2+p(2,1)*q(2,1)<0,eN+M2^2+(p(3,1)+q(3,1))*M
2+p(3,1)*q(3,1)<0];
obj=eN;
EN=solvesdp(H,-obj);
en=double(eN);
checkset(H);

eX=intvar(1,1);
J=[eX+M2^2+(p(1,1)+q(1,1))*M2+p(1,1)*q(1,1)>0,eX+M2^2+(p(
2,1)+q(2,1))*M2+p(2,1)*q(2,1)<0,eX+M2^2+(p(3,1)+q(3,1))*M
2+p(3,1)*q(3,1)>0];
obje=eX;
EX=solvesdp(J,-obje);
ex=double(eX);
checkset(J);
```

```
e=(ex+en)/2;

M1=[1 M2;M2 (M2^2+e)];

for k=1:N-1
M(2*k-1:2*k,:)=T(2*k-1:2*k,:)'*M1*T(2*k-1:2*k,:);
end

M=[M1;M];

for k=1:N
Ao(2*k-1:2*k,:)=C(2*k-1:2*k,:)*A(2*k-1:2*k,:)*inv(C(2*k-
1:2*k,:));
end

for k=1:N
Ko(k,:)=[-1-Ao((2*k),1) (((((-1)*M((2*k),2))-M((2*k-
1),1))/M((2*k-1),2))-Ao(2*k,2))];
end

for k=1:N
K(k,:)=Ko(k,:)*C(2*k-1:2*k,:)
end

Mo=C(1:2,:)'*M(1:2,:)*C(1:2,:)

for k=1:N
P(2*k-1:2*k,:)=(Mo(1:2,:)*(A(2*k-1:2*k,:)+(b(2*k-
1:2*k,:)*K(k,:))))+(((A(2*k-1:2*k,:)+(b(2*k-
1:2*k,:)*K(k,:)))')*Mo(1:2,:));
end

for k=1:N
```

```
    Z(2*k-1:2*k,:)=eig(P(2*k-1:2*k,:));
end

Z
```

OUTPUT (SeDuMi-1.3 and LINPROG)

```
* Starting YALMIP integer branch & bound.
* Lower solver   : SeDuMi-1.3
* Upper solver   : rounder
* Max iterations : 300
 Node        Upper        Gap(%)      Lower    Open
    1 :   3.000E+000      0.00     3.000E+000   0  No
problems detected
+   1 Finishing.  Cost: 3

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
|   ID|      Constraint|
Type|   Primal residual|
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
|   #1|   Numeric value|              Elementwise
inequality|                3|
|   #2|   Numeric value|              Elementwise
inequality|                3|
|   #3|   Numeric value|   Elementwise inequality
(quadratic)|                2|
|   #4|   Numeric value|   Elementwise inequality
(quadratic)|                2|
|   #5|   Numeric value|   Elementwise inequality
(quadratic)|       6.3949e-014|
```

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++

* Starting YALMIP integer branch & bound.
* Lower solver   : SeDuMi-1.3
* Upper solver   : rounder
* Max iterations : 300
 Node        Upper       Gap(%)       Lower    Open
    1 :  -4.000E+000      0.00    -4.000E+000    0  No
problems detected
+   1 Finishing.  Cost: -4

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
|   ID|      Constraint|
Type|   Primal residual|
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
|   #1|   Numeric value|              Elementwise
inequality|                4|
|   #2|   Numeric value|              Elementwise
inequality|                4|
|   #3|   Numeric value|   Elementwise inequality
(quadratic)|       7.1054e-015|
|   #4|   Numeric value|   Elementwise inequality
(quadratic)|                2|
|   #5|   Numeric value|   Elementwise inequality
(quadratic)|                2|
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++

* Starting YALMIP integer branch & bound.
* Lower solver   : LINPROG
```

```
* Upper solver   : rounder
* Max iterations : 300
 Node       Upper       Gap(%)      Lower    Open
    1 :  -1.000E+000     0.00    -1.000E+000   0  No
problems detected
+   1 Finishing.  Cost: -1

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++
|   ID|      Constraint|                    Type|
Primal residual|
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++
|   #1|   Numeric value|   Elementwise inequality|
0.25|
|   #2|   Numeric value|   Elementwise inequality|
1.25|
|   #3|   Numeric value|   Elementwise inequality|
0.25|
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++

* Starting YALMIP integer branch & bound.
* Lower solver   : LINPROG
* Upper solver   : rounder
* Max iterations : 300
 Node       Upper       Gap(%)      Lower    Open
    1 :  -2.000E+000     0.00    -2.000E+000   0  No
problems detected
+   1 Finishing.  Cost: -2

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++
```

```
|   ID|      Constraint|                        Type|
Primal residual|
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++
|   #1|   Numeric value|   Elementwise inequality|
0.75|
|   #2|   Numeric value|   Elementwise inequality|
0.25|
|   #3|   Numeric value|   Elementwise inequality|
0.75|
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++

K =
 -11.214285714285715  -8.214285714285715
 -36.000000000002139 -23.333333333334668
 -18.000000000000043 -13.500000000000030
  58.000000000014730  46.000000000011724

Mo =
  31.750000000000000  26.250000000000000
  26.250000000000000  21.750000000000000

Z =
  1.0e+003 *
  -0.417771352362701
  -0.000014361923014
  -0.172965310963419
  -0.000034689036586
  -0.297479830565360
  -0.000020169434643
  -2.452997554013643
  -0.000002445986950
```

OUTPUT (LMILAB and LINRPOG)

```
* Starting YALMIP integer branch & bound.
* Lower solver   : LMILAB
* Upper solver   : rounder
* Max iterations : 300
 Node        Upper        Gap(%)       Lower     Open
    1 :   0.000E+000      0.00     0.000E+000    0  No
problems detected
+   1 Finishing.  Cost: 0

++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
|   ID|       Constraint|
Type|   Primal residual|
++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
|   #1|   Numeric value|                Elementwise
inequality|                  0|
|   #2|   Numeric value|                Elementwise
inequality|                  0|
|   #3|   Numeric value|   Elementwise inequality
(quadratic)|                 -4|
|   #4|   Numeric value|   Elementwise inequality
(quadratic)|                -10|
|   #5|   Numeric value|   Elementwise inequality
(quadratic)|                -18|
++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++

* Starting YALMIP integer branch & bound.
* Lower solver   : LMILAB
* Upper solver   : rounder
```

```
* Max iterations : 300
 Node       Upper       Gap(%)      Lower    Open
    1 :           Inf      Inf    -1.000E+001   2  No
problems detected
    2 :           Inf      Inf    -1.000E+001   1
Infeasible problem
    3 :           Inf      Inf    -9.000E+000   2  No
problems detected
    4 :           Inf      Inf    -9.000E+000   1
Infeasible problem
    5 :  -8.000E+000     0.00    -8.000E+000   0  No
problems detected
+   5 Finishing.  Cost: -8

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
|   ID|      Constraint|
Type|   Primal residual|
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
|   #1|   Numeric value|              Elementwise
inequality|          8|
|   #2|   Numeric value|              Elementwise
inequality|           2|
|   #3|   Numeric value|   Elementwise inequality
(quadratic)|          -28|
|   #4|   Numeric value|   Elementwise inequality
(quadratic)|          -18|
|   #5|   Numeric value|   Elementwise inequality
(quadratic)|          -10|
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
```

```
* Starting YALMIP integer branch & bound.
* Lower solver   : LINPROG
* Upper solver   : rounder
* Max iterations : 300
 Node       Upper       Gap(%)      Lower    Open
    1 :   0.000E+000     0.00     0.000E+000   0  No
problems detected
+   1 Finishing.  Cost: 0

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++
|   ID|      Constraint|                    Type|
Primal residual|
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++
|   #1|   Numeric value|   Elementwise inequality|
5.3291e-015|
|   #2|   Numeric value|   Elementwise inequality|
2|
|   #3|   Numeric value|   Elementwise inequality|
2|
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++

* Starting YALMIP integer branch & bound.
* Lower solver   : LINPROG
* Upper solver   : rounder
* Max iterations : 300
 Node       Upper       Gap(%)      Lower    Open
    1 :          Inf      Inf            NaN   0
Infeasible problem
+   1 Finishing.  Cost: Inf
```

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++
|   ID|      Constraint|                        Type|
Primal residual|
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++
|   #1|   Numeric value|   Elementwise inequality|
2|
|   #2|   Numeric value|   Elementwise inequality|      -
1.7764e-015|
|   #3|   Numeric value|   Elementwise inequality|      -
4.9738e-014|
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++

K =

 -11.500000000000000  -8.500000000000000
  -8.999999999999968  -5.333333333333286
 -16.000000000000028 -12.000000000000020
 -16.999999999999442 -13.999999999999545

Mo =
    37    31
    31    26

Z =
  1.0e+002 *
  -5.339925092582186
  -0.000074907417813
  -0.298660687473191
  -0.001339312526813
  -1.409716254956561
```

```
-0.000283745043440
 0.000082305920629
 4.859917694079162
```

B.4 Maple

```
> with(LinearAlgebra):
> for n from 1 to k do AB[n] := A[n].B[n];end do:
> for n from 1 to k do ABi[n] := AB[n]-(((<B[n]|AB[n]>)^-1.(A[n]
      .AB[n]))[2].B[n]);end do:
> for n from 1 to k do C[n] := (<ABi[n]|B[n]>)^-1;end do
```

$$C_1 := \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}$$

$$C_2 := \begin{bmatrix} -2 & -\frac{5}{3} \\ 1 & \frac{2}{3} \end{bmatrix}$$

$$C_3 := \begin{bmatrix} \frac{7}{6} & 1 \\ \frac{4}{3} & 1 \end{bmatrix}$$

$$C_4 := \begin{bmatrix} \frac{8}{3} & \frac{7}{3} \\ -\frac{5}{3} & -\frac{4}{3} \end{bmatrix}$$

```
> for n from 1 to k - 1 do T[n] := C[1].C[n + 1]^-1;end do
```

$$T_1 := \begin{bmatrix} 1 & 4 \\ -1 & -1 \end{bmatrix}$$

$$T_2 := \begin{bmatrix} -4 & 5 \\ 2 & -1 \end{bmatrix}$$

$$T_3 := \begin{bmatrix} -3 & -6 \\ 1 & 1 \end{bmatrix}$$

```
> for n from 1 to k - 1 do  Z[n] := T[n][2, 1].T[n][2, 2];end do:
> for n from 1 to k - 1 do if Z[n] > 0 then print(Sp) elif Z[n] < 0
      then print(Sn) elif Z[n] = 0 then print(Sz) end if;end do;
```

Sp

$$Sn$$

$$Sp$$

> **for** n **from** 1 **to** $k-1$ **do** $PSz[n] := T[n][1,2]*T[n][2,1] + T[n][1,1]*T[n][2,2]$;**end do**

$$PSz_1 := -5$$

$$PSz_2 := 14$$

$$PSz_3 := -9$$

> **for** n **from** 1 **to** $k-1$ **do** $QSz[n] := T[n][1,1]*T[n][1,2]$; **end do**

$$QSz_1 := 4$$

$$QSz_2 := -20$$

$$QSz_3 := 18$$

> **for** n **from** 1 **to** $k-1$ **do** $P[n] := \frac{T[n][1,1]}{T[n][2,1]}$; **end do**

$$P_1 := -1$$

$$P_2 := -2$$

$$P_3 := -3$$

> **for** n **from** 1 **to** $k-1$ **do** $Q[n] := \frac{T[n][1,2]}{T[n][2,2]}$; **end do**

$$Q_1 := -4$$

$$Q_2 := -5$$

$$Q_3 := -6$$

> $with(Optimization)$:

> $with(plots)$:

> $obj := yy$

$$obj := yy$$

> **for** n **from** 1 **to** $k-1$ **do if** $Z[n] > 0$ **then** $RR[n] := (xx^2 + (P[n] + Q[n])*xx + (P[n]*Q[n]) \geq yy)$ **elif** $Z[n] < 0$ **then** $RR[n] := (xx^2 + (P[n] + Q[n])*xx + (P[n]*Q[n]) \leq yy)$ **elif** $Z[n] = 0$ **then** $RR[n] := (PSz[n]*xx + QSz[n] \geq yy)$ **end if end do**

```
> for n from 1 to k - 1 do if n = 1 then cnsts[n] := [RR[n], 0 ≤ xx]
    elif n = 2 then cnsts[n] := [RR[n - 1], RR[n], 0 ≤ xx] elif n
    = 3 then cnsts[n] := [RR[n - 2], RR[n - 1], RR[n], 0 ≤ xx]
    elif n = 4then cnsts[n] := [RR[n - 3], RR[n - 2], RR[n - 1],
    RR[n], 0 ≤ xx] elif n = 5 then cnsts[n] := [RR[n - 4], RR[n
    - 3], RR[n - 2], RR[n - 1], RR[n], 0 ≤ xx] elif n = 6
    then cnsts[n] := [RR[n - 5], RR[n - 4], RR[n - 3], RR[n
    - 2], RR[n - 1], RR[n], 0 ≤ xx] elif n = 7 then cnsts[n]
    := [RR[n - 6], RR[n - 5], RR[n - 4], RR[n - 3], RR[n - 2],
    RR[n - 1], RR[n], 0 ≤ xx] elif n = 8 then cnsts[n] := [RR[n
    - 7], RR[n - 6], RR[n - 5], RR[n - 4], RR[n - 3], RR[n
    - 2], RR[n - 1], RR[n], 0 ≤ xx] elif n = 9then cnsts[n]
    := [RR[n - 8], RR[n - 7], RR[n - 6], RR[n - 5], RR[n - 4],
    RR[n - 3], RR[n - 2], RR[n - 1], RR[n], 0 ≤ xx] elif n = 10
    then cnsts[n] := [RR[n - 9], RR[n - 8], RR[n - 7], RR[n
    - 6], RR[n - 5], RR[n - 4], RR[n - 3], RR[n - 2], RR[n
    - 1], RR[n], 0 ≤ xx] end if end do

> cnsts[3]

    [yy ≤ xx^2 - 5 xx + 4, xx^2 - 7 xx + 10 ≤ yy, yy ≤ xx^2 - 9 xx + 18,
     0 ≤ xx]

> elow := NLPSolve(obj, cnsts[k - 1], xx = 0..10)

    elow := [-2.250000000000019540, [xx = 3.499999772578381 12, yy =
     -2.250000000000019540]]

> vx := rhs(elow[2][1])

                    vx := 3.4999997725783811

> eobj := e

                          eobj := e

> for n from 1 to k - 1 do if Z[n] > 0 then e1[n] := (e + vx^2
    + (P[n] + Q[n])vx + (P[n]*Q[n]) ≥ vy) elif Z[n] < 0
    then e1[n] := (e + vx^2 + (P[n] + Q[n])vx + (P[n]*Q[n])
    ≤ vy) elif Z[n] = 0 then e1[n] := ((e + ((PSz[n])*vx
    + QSz[n])) ≥ 0) end if end do
```

```
> for n from 1 to k − 1 do if n = 1 then ecnsts[n] := [e1[n], 0 = vy]
    elif n = 2 then ecnsts[n] := [e1[n − 1], e1[n], 0 = vy] elif n = 3
    then ecnsts[n] := [e1[n − 2], e1[n − 1], e1[n], 0 = vy] elif n
    = 4then ecnsts[n] := [e1[n − 3], e1[n − 2], e1[n − 1], e1[n],
    0 = vy] elif n = 5 then ecnsts[n] := [e1[n − 4], e1[n − 3], e1[n
    − 2], e1[n − 1], e1[n], 0 = vy] elif n = 6 then ecnsts[n]
    := [e1[n − 5], e1[n − 4], e1[n − 3], e1[n − 2], e1[n − 1],
    e1[n], 0 = vy] elif n = 7 then ecnsts[n] := [e1[n − 6], e1[n
    − 5], e1[n − 4], e1[n − 3], e1[n − 2], e1[n − 1], e1[n], 0
    = vy] elif n = 8 then ecnsts[n] := [e1[n − 7], e1[n − 6], e1[n
    − 5], e1[n − 4], e1[n − 3], e1[n − 2], e1[n − 1], e1[n], 0
    = vy] elif n = 9 then ecnsts[n] := [e1[n − 8], e1[n − 7], e1[n
    − 6], e1[n − 5], e1[n − 4], e1[n − 3], e1[n − 2], e1[n − 1],
    e1[n], 0 = vy] elif n = 10 then ecnsts[n] := [e1[n − 9], e1[n
    − 8], e1[n − 7], e1[n − 6], e1[n − 5], e1[n − 4], e1[n − 3],
    e1[n − 2], e1[n − 1], e1[n], 0 = vy] end if end do
```

```
> ecnsts[3]
```

$$[vy \le e - 1.25000045, e - 2.25000000 \le vy, vy \le e - 1.24999955, 0 = vy]$$

```
> low := LPSolve(eobj, ecnsts[k − 1], e = 0..10, vy = 0..10)
```

$$low := [1.25000045000000, [e = 1.25000044999999994, vy = 0.]]$$

```
> elow := rhs(low[2][1])
```

$$elow := 1.2500004499999999$$

```
> high := LPSolve(eobj, ecnsts[k − 1], e = 0..10, vy = 0..10,
    maximize)
```

$$high := [2.25000000000000, [e = 2.25000000000000000, vy = 0.]]$$

```
> ehigh := rhs(high[2][1])
```

$$ehigh := 2.2500000000000000$$

```
> ve := (ehigh − elow)/2 + elow
```

$$ve := 1.75000022:$$

```
> M[1] := Matrix([[1, vx], [vx, vx^2 + ve]])
```

$$M_1 := \begin{bmatrix} 1 & 3.49999977257838112 \\ 3.49999977257838112 & 13.99999864 \end{bmatrix}$$

```
> for n from 1 to k − 1 do  M[n + 1] := T[n]^%T · (M[1] · T[n]) :
    end do:
```

```
> for n from 1 to k do if M[n][2, 1] > 0 then EM[n]
    := Eigenvalues (evalf (M[n])) else EM[n] := [ 0 ]
                                                [ 0 ] end if;end
  do
```

```
> for n from 1 to k do if EM[n][1] > 0 and EM[n][2] > 0
  and M[n][2, 1] > 0 then S[n] := C[n] . (A[n] . C[n]^-1)
  else S[n] := [ 0 0 ]
               [ 0 0 ] end if;end do
```

```
> for n from 1 to k do if EM[n][1] > 0 and EM[n][2] > 0
  and M[n][2, 1] > 0 then ko[n]
  := [(-1 - S[n][2, 1]), ( (-1*M[n][2, 2]-M[n][1, 1]) / M[n][1, 2] - S
  [n][2, 2])] else ko[n] := [ 0 0 ] end if;end do
```

```
> for n from 1 to k do if EM[n][1] > 0 and EM[n][2] > 0
  and M[n][2, 1] > 0 then K[n] := ko[n] . C[n]; print(K
  = K[n]) else K[n] := [ 0 0 ]; print(K = K[n]) end if;end do
```

$$K = \begin{bmatrix} -11.2857141799999994 & -8.2857141749999896 \end{bmatrix}$$

$$K = \begin{bmatrix} -18.0000080300000000 & -11.3333386899999997 \end{bmatrix}$$

$$K = \begin{bmatrix} -28.0000113100000014 & -21.0000084800000018 \end{bmatrix}$$

$$K = \begin{bmatrix} 27.999959140000001421.9999673000000016 \end{bmatrix}$$

```
> M[0] := C[1]^%T . (M[1] . C[1])
```

$$M_0 := \begin{bmatrix} 31.999997735999997426.4999979560000014 \\ 26.499997962999998421.9999981829999988 \end{bmatrix}$$

```
> for n from 1 to k do if EM[n][1] > 0 and EM[n][2]
    > 0 and M[n][2, 1] > 0 then E[n] := (M[0]
    . (A[n] + (B[n] . K[n]))) + ((A[n] + (B[n]
    . K[n]))^%T . M[0]) else E[n] := [ 1 1 ]
                                    [ 1 1 ] end if;
  end do:
```

```
> for n from 1 to k do Ee[n] := (evalf(E[n])) end do
```

$$Ee_1 := \begin{bmatrix} -252.999973199999999 & -210.999976599999997 \\ -210.999976699999991 & -175.999979700000012 \end{bmatrix}$$

$$Ee_2 := \begin{bmatrix} -67.0000461000000058 & -44.0000302300000001 \\ -44.0000303699999975 & -29.0000198500000010 \end{bmatrix}$$

$$Ee_3 := \begin{bmatrix} -308.500193299999978 & -233.000145300000014 \\ -233.000145500000002 & -176.000109400000014 \end{bmatrix}$$

$$Ee_4 := \begin{bmatrix} -750.999044700000013 & -595.999235600000020 \\ -595.999235799999952 & -472.999388500000009 \end{bmatrix}$$

```
> for n from 1 to k do E[n]
      := Eigenvalues(evalf(E[n])) end do
```

$$E_1 := \begin{bmatrix} -428.983635258895220 + 0.\,I \\ -0.0163176411048482351 + 0.\,I \end{bmatrix}$$

$$E_2 := \begin{bmatrix} -95.9270938607990616 + 0.\,I \\ -0.0729720892009417810 + 0.\,I \end{bmatrix}$$

$$E_3 := \begin{bmatrix} -484.485854364229340 + 0.\,I \\ -0.0144483357706803872 + 0.\,I \end{bmatrix}$$

$$E_4 := \begin{bmatrix} -1223.99271425076222 + 0.\,I \\ -0.00571894923763238694 + 0.\,I \end{bmatrix}$$

```
> for n from 1 to k do if E[n][1] < 0 and E[n][2] < 0
      and EM[n][1] > 0 and EM[n][2] > 0
      and M[n][2, 1] > 0 then l[n] = 0
      and printf("Subsystem %a Stable \n", n) else l[n] = 1
      and printf("Subsystem %a Not Stable \n", n) end if;end
      do;
```

```
Subsystem 1 Stable
Subsystem 2 Stable
Subsystem 3 Stable
Subsystem 4 Stable
```

```
> for n from 1 to k do if E[n][1] < 0 and E[n][2] < 0 then l[n] := 0
      else l[n] := 1 end if;end do;
```

```
> X := add(l[n], n = 1..k):
```

```
> if X = 0 then print("System Stable") else print("System not Stable")
      end if;
```

"System Stable"

APPENDIX C

CASE STUDY 2

INPUT

```
A1=[15 22
    35 40]
A2=[11 31
    38 42]
A3=[10 33
    51 64]
A4=[17 63
    52 64]
B1=[12
    23]
B2=[12
    15]
B3=[14
    20]
B4=[15
    16]
I=[1 0;0 1]
```

OUTPUT

METHOD *M*

C.1 LMI Solver

```
M =
   0.581243990292193   0.620561867726289
   0.344025192363398   0.828583200816174

Z1 =
  -1.416870008789574  -2.413752578039924
Z2 =
  -1.467315107948165  -3.930684366093832
Z3 =
  -1.272727773704987  -4.265964563400459
Z4 =
  -2.147498753769431  -5.372059236201160

K1 =
  -1.281526468337807  -1.953317563900998
K2 =
   0.508957745048945  -5.125042519293467
K3 =
   1.540497868004962  -6.302250386508264
K4 =
   0.256386311712721  -6.675446472012394

ans =
  -2.249964696323282
  -2.413114660175904
ans =
  -4.443422072043312
```

```
 -18.866136573639242
ans =
  -7.500141466641082
 -32.985306520356232
ans =
  -5.589788413049607
 -22.817102574743828
```

C.2 Yalmip

```
Z1_feasible =
  -1.609070423476548  -2.570519977095403
Z2_feasible =
  -1.828196083902233  -4.175130517072147
Z3_feasible =
  -1.618043782536687  -4.583558661135924
Z4_feasible =
  -2.776750061987877  -5.763501203993872

M_feasible =
   0.487831217172648   0.519079074155418
   0.519079074155418   1.008969459127232

K1 =
  -1.298235604570850  -1.879772498842307
K2 =
   1.448300817821444  -4.883114270819720
K3 =
   3.351859296440412  -6.267225062393149
K4 =
   0.853176925365393  -6.151194603919802
```

```
ans =
  -1.208612947791101
  -1.125488462958981
ans =
  -1.208612947791080
  -0.803375000093562
ans =
  -1.328959838824267
  -1.208612947791133
ans =
  -1.403388231747367
  -1.208612947791068
```

METHOD *H*

OUTPUT

C.3 cvx

Infeasible Solution

C.4 Yalmip (LMILAB and LINPROG)

```
K =
  -1.256622516556304  -1.844370860927155
   0.145300822382341  -2.444444219680717
   0.800750618869548  -1.663468587445111
  -0.054710244430580  -4.303885646195536
```

```
Mo =
    0.409565369939858  -0.129753081005195
   -0.129753081005195   0.040919257927274

Z =
   -1.811372942940327
    0.000169486951853
   -0.011316339974432
    0.027129273198058
   -0.031579948052716
    0.009721487769898
   -0.021781320364839
    0.014094833261990
```

C.5 Yalmip (SeDuMi-1.3 and LINPROG)

```
K =
   -1.355960264900668  -1.809933774834443
    0.180329303097957  -2.439749681028109
    0.855093225102965  -1.638517507540273
   -0.020421346705987  -4.307125542043528

Mo =
    0.649275251085391  -0.209541248190832
   -0.209541248190832   0.067431253015204
Z =
   -3.586499779346815
    0.000140628432935
   -0.009744947863226
    0.051756443520270
   -0.030541520169824
    0.016514038623052
```

```
-0.011382245608383
 0.044311453209172
```

C.6 Maple

```
> for n from 1 to k do if EM[n][1] > 0 and EM[n][2] > 0
    and M[n][2, 1] > 0 then K[n] := ko[n] • C[n]; print(K
    = K[n]) else K[n] := [ 0 0 ]; print(K = K[n]) end if;end do
```

$$K = \begin{bmatrix} -1.63719849000000006 & -1.71243785499999990 \end{bmatrix}$$

$$K = \begin{bmatrix} -8.24485926499999877 & -3.56889866499999986 \end{bmatrix}$$

$$K = \begin{bmatrix} -8.74218690900000084 & -6.04505079900000020 \end{bmatrix}$$

$$K = \begin{bmatrix} -10.9013831900000006 & -3.27900316400000014 \end{bmatrix}$$

$$M[0] := C[1]^{\%T} \cdot (M[1] \cdot C[1])$$

$$M_0 := \begin{bmatrix} 0.00046422989801324504 & -0.00029172008109271525 \\ -0.00029172008033112584 & 0.00032424691717880798 \end{bmatrix}$$

for n **from** 1 **to** k **do** $E[n]$ $:= Eigenvalues(evalf(E[n]))$ **end do**

$$E_1 := \begin{bmatrix} -0.00312928039976713511 + 0.\,I \\ -0.00008362880653286510 - 0.\,I \end{bmatrix}$$

$$E_2 := \begin{bmatrix} -0.03223323184803039700 + 0.\,I \\ -0.00000811889006960351666 - 0.\,I \end{bmatrix}$$

$$E_3 := \begin{bmatrix} -0.03886455061061233622 + 0.\,I \\ -0.00000673359238765811053 - 0.\,I \end{bmatrix}$$

$$E_4 := \begin{bmatrix} -0.06518782493144541233 + 0.\,I \\ -0.00000401452475460152645 - 0.\,I \end{bmatrix}$$

```
> for n from 1 to k do if E[n][1] < 0 and E[n][2] < 0
    and EM[n][1] > 0 and EM[n][2] > 0
    and M[n][2, 1] > 0 then l[n] = 0
    and  printf ("Subsystem %a Stable \n", n) else l[n] = 1
    and printf ("Subsystem %a Not Stable \n", n) end if;end
    do;
```

```
Subsystem 1 Stable
Subsystem 2 Stable
Subsystem 3 Stable
Subsystem 4 Stable
```

```
> for n from 1 to k do if E[n][1] < 0 and E[n][2] < 0 then l[n] := 0
    else  l[n] := 1 end if;end do;
```

```
> X := add (l[n], n = 1 ..k) :
```

```
> if X = 0 then print ("System Stable") else  print ("System not Stable")
    end if;
```

```
"System Stable"
```

APPENDIX D

CASE STUDY 3

INPUT

```
A1=[0 1
    -3 -0.2]
A2=[-0.2 -3
    1 0]
B1=[0
    1]
B2=[1
    0]
```

METHOD *M*

OUTPUT

D.1 LMI Solver

```
M =
  81.910256511916018 -32.764102604766407
 -32.764102604766386  81.910256511916103

Z1 =
  1.0e+002 *
   1.572676925028786  -1.228653847678740
```

```
Z2 =
  1.0e+002 *
  -1.228653847678739   1.572676925028791
K1 =
   1.571428571428570  -0.871428571428570
K2 =
  -0.871428571428569   1.571428571428574

ans =
 -65.528205209532786
 -81.910256511915961
ans =
 -81.910256511915875
 -65.528205209532786
```

D.2 Yalmip

```
Z1_feasible =
  1.0e+002 *
   1.572676925028767  -1.228653847678725
Z2_feasible =
  1.0e+002 *
  -1.228653847678726   1.572676925028782

M_feasible =
  81.910256511915165 -32.764102604766016
 -32.764102604766016  81.910256511915534

K1 =
   1.571428571428569  -0.871428571428567
K2 =
  -0.871428571428568   1.571428571428580
```

```
ans =
 -81.910256511915193
 -65.528205209532032
ans =
 -81.910256511915236
 -65.528205209532032
```

METHOD *H*

OUTPUT

D.3 Maple

```
> for n from 1 to k do if EM[n][1] > 0 and EM[n][2] > 0
    and M[n][2, 1] > 0 then K[n] := ko[n] • C[n]; print(K
    = K[n]) else K[n] := [ 0  0 ]; print(K = K[n]) end if;end do
```

$$K = \begin{bmatrix} 2. & -6. \end{bmatrix}$$

$$K = \begin{bmatrix} -6. & 2. \end{bmatrix}$$

$$M[0] := C[1]^{\%T} \cdot (M[1] \cdot C[1])$$

$$M_0 := \begin{bmatrix} 1. & 5. \\ 5. & 30. \end{bmatrix}$$

```
for n from 1 to k do E[n]
    := Eigenvalues(evalf(E[n])) end do
```

$$E_1 := \begin{bmatrix} -0.0537712132886554174 + 0.\,I \\ -371.946228786711345 + 0.\,I \end{bmatrix}$$

$$E_2 := \begin{bmatrix} -1.90581788928322382 + 0.\,I \\ -10.4941821107167784 + 0.\,I \end{bmatrix}$$

APPENDIX E

CASE STUDY 4

INPUT

```
A1=[-0.1 1
    -10 -0.1]
A2=[-0.1 10
    -1 -0.1]
B1=[1
    0]
B2=[1
    0]
```

METHOD *M*

OUTPUT

E.1 LMI Solver

```
M =
   1.719766468732211   0.072591830990231
  -0.072591830990235   1.933582407285310

Z1 =
  -0.615314756502654  15.264082280036799
Z2 =
   0.038011722409463 -17.616057604120893
```

```
K1 =
  -0.024534499351431   7.895119042637520
K2 =
  -0.361883865433702  -9.096994121092960

ans =
  -0.420591516742975
  -0.394465477083971
ans =
  -1.587854226542870
  -0.387527023356795
```

E.2 Yalmip

```
Z1_feasible =
  -0.800232686590474  16.083101140625939
Z2_feasible =
  -1.484487618119711 -18.546418894133158
M_feasible =
   1.810510902162492   0.076028325725471
   0.076028325725471   2.037213546144074
K1 =
  -0.774725174761981   7.923569048081954
K2 =
  -0.438320111832123  -9.087458791416013

ans =
  -1.928009223738231
  -1.810510902162504
ans =
  -1.810510902162504
  -0.559499360679757
```

METHOD *H*

E.2 Maple

```
> for n from 1 to k do if EM[n][1] > 0 and EM[n][2] > 0
    and M[n][2, 1] > 0 then K[n] := ko[n] • C[n]; print(K
    = K[n]) else K[n] := [ 0  0 ]; print(K = K[n]) end if;end do
```

$$K = \begin{bmatrix} -6. & -0.960999999999999965 \end{bmatrix}$$

$$K = \begin{bmatrix} -9.28218749999999914 & -9.93821875000000077 \end{bmatrix}$$

$$M[0] := C[1]^{\%T} \bullet (M[1] \bullet C[1])$$

$$M_0 := \begin{bmatrix} 30. & -0.200000000000000010 \\ -0.199999999999999954 & 0.00300000000000000006 \end{bmatrix}$$

```
for n from 1 to k do E[n]
    := Eigenvalues(evalf(E[n])) end do
```

$$E_1 := \begin{bmatrix} -362.015647537692473 + 0.\,I \\ -0.000552462307528232139 + 0.\,I \end{bmatrix}$$

$$E_2 := \begin{bmatrix} -562.556206979969261 + 0.\,I \\ -0.000355520030742670358 + 0.\,I \end{bmatrix}$$